Methods for 3D Geometry Processing in the Cultural Heritage Domain

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Dipl.-Math. Gerhard Heinrich Bendels

Bonn, Dezember 2006

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms Universität Bonn

Dekan: Prof. Dr. Matthias Winiger

1. Referent: Prof. Dr. Reinhard Klein
2. Referent: Prof. Dr. Stefan Gumhold

Tag der Promotion: 30.05.2007
Erscheinungsjahr: 2007
Diese Dissertation ist auf dem Hochschulschriftenserver der ULB Bonn
`http://hss.ulb.uni-bonn.de/diss_online`
elektronisch publiziert

Contents

Preface

Abstracting information from its original medium or carrier to make its properties visible, understandable, or accessible in another manner is one of the fundamental concepts of scientific research. The concrete form or method of abstraction varies in time, in reply to the field of research and the scientific question at hand. It is also naturally influenced by the technological progress, as new technology has always spawned new research methodologies and led to the formation and validation of new scientific hypotheses.

The one technological development that originated and motivated much of the research presented in this thesis is *3D Photography* – the generation of digital copies of three-dimensional objects. While this technology has a vast impact in various fields of automation, manufacturing and reverse engineering, the focus of the present work is its use – and the methods necessitated by its use – in the context of cultural heritage applications.

In the cultural heritage sector, the use of virtual, yet faithful reconstructions of valuable artefacts can break ground for numerous fascinating applications that were not within possibility before, given the various severely limiting restrictions that characterise the handling of their physical counterparts. On the other hand, applying computer graphics technologies in this sector also poses several special demands, such as dependability of the data, as well as the differentiation between acquired, reconstructed and guessed data.

Before this background, the present thesis investigates methods for acquiring, reconstructing and recreating virtual representations of cultural heritage artefacts, focussing on fully automatic methods wherever possible and on intuitive and easy to use interaction paradigms where necessary to take into account that future users of the presented methods will most likely be experts in a field different from computer science.

Abstract

This thesis presents methods for 3D geometry processing under the aspects of cultural heritage applications. After a short overview over the relevant basics in 3D geometry processing, the present thesis investigates the digital acquisition of 3D models. A particular challenge in this context are on the one hand difficult surface or material properties of the model to be captured. On the other hand, the fully automatic reconstruction of models even with suitable surface properties that can be captured with Laser-range scanners is not yet completely solved. This thesis presents two approaches to tackle these challenges. One exploits a thorough capture of the object's appearance and a coarse reconstruction for a concise and realistic object representation even for objects with problematic surface properties like reflectivity and transparency. The other method concentrates on digitisation via Laser-range scanners and exploits 2D colour images that are typically recorded with the range images for a fully automatic registration technique.

After reconstruction, the captured models are often still incomplete, exhibit holes and/or regions of insufficient sampling. In addition to that, holes are often deliberately introduced into a registered model to remove some undesired or defective surface part. In order to produce a visually appealing model, for instance for visualisation purposes, for prototype or replica production, these holes have to be detected and filled. Although completion is a well-established research field in 2D image processing and many approaches do exist for image completion, surface completion in 3D is a fairly new field of research. This thesis presents a hierarchical completion approach that employs and extends successful exemplar-based 2D image processing approaches to 3D and fills-in detail-equipped surface patches into missing surface regions.

In order to identify and construct suitable surface patches, self-similarity and coherence properties of the surface context of the hole are exploited.

In addition to the reconstruction and repair, the present thesis also investigates methods for a modification of captured models via interactive modelling. In this context, modelling is regarded as a creative process, for instance for animation purposes. On the other hand, it is also demonstrated how this creative process can be used to introduce human expertise into the otherwise automatic completion process. This way, reconstructions are feasible even of objects where already the data source, *the object itself*, is incomplete due to corrosion, demolition, or decay.

Zusammenfassung

In dieser Arbeit werden Methoden zur Bearbeitung von digitaler 3D Geometrie unter besonderer Berücksichtigung des Anwendungsbereichs im Kulturerbesektor vorgestellt. Nach einem kurzen Überblick über die relevanten Grundlagen der dreidimensionalen Geometriebehandlung wird zunächst die digitale Akquise von dreidimensionalen Objekten untersucht. Eine besondere Herausforderung stellen bei der Erfassung einerseits ungünstige Oberflächen- oder Materialeigenschaften der Objekte dar (wie z.B. Reflexivität oder Transparenz), andererseits ist auch die vollautomatische Rekonstruktion von solchen Modellen, die sich verhältnismäßig problemlos mit Laser-Range Scannern erfassen lassen, immer noch nicht vollständig gelöst. Daher bilden zwei neuartige Verfahren, die diesen Herausforderungen begegnen, den Anfang.

Auch nach der Registrierung sind die erfassten Datensätze in vielen Fällen unvollständig, weisen Löcher oder nicht ausreichend abgetastete Regionen auf. Darüber hinaus werden in vielen Anwendungen auch, z.B. durch Entfernen unerwünschter Oberflächenregionen, Löcher gewollt hinzugefügt. Für eine optisch ansprechende Rekonstruktion, vor allem zu Visualisierungszwecken, im Bildungs- oder Unterhaltungssektor oder zur Prototyp- und Replik-Erzeugung müssen diese Löcher zunächst automatisch detektiert und anschließend geschlossen werden. Obwohl dies im zweidimensionalen Fall der Bildbearbeitung bereits ein gut untersuchtes Forschungsfeld darstellt und vielfältige Ansätze zur automatischen Bildvervollständigung existieren, ist die Lage im dreidimensionalen Fall anders, und die Übertragung von zweidimensionalen Ansätzen in den 3D stellt vielfach eine große Herausforderung dar, die bislang keine zufriedenstellenden Lösungen erlaubt hat. Nichtsdestoweniger wird in dieser Arbeit ein hierarchisches Verfahren vorgestellt, das beispielbasierte Konzepte aus dem 2D aufgreift und Löcher in Oberflächen im 3D unter Ausnutzung von Selbstähnlichkeiten und Kohärenzeigenschaften des Oberflächenkontextes schließt. Um plausible Oberflächen zu erzeugen werden die Löcher dabei nicht nur glatt gefüllt, sondern auch feinere Details aus dem Kontext rekonstruiert.

Abschließend untersucht die vorliegende Arbeit noch die Modifikation der vervollständigten Objekte durch Freiformmodellierung. Dies wird dabei zum einen als kreativer Prozess z.B. zu Animationszwecken betrachtet. Zum anderen wird aber auch untersucht, wie dieser kreative Prozess benutzt werden kann, um etwaig vorhandenes Expertenwissen in die ansonsten automatische Vervollständigung mit einfließen zu lassen. Auf diese Weise werden auch Rekonstruktionen ermöglicht von Objekten, bei denen schon die *Datenquelle*, also das Objekt selbst z.B. durch Korrosion oder mutwillige Zerstörung unvollständig ist.

Acknowledgements

Many have contributed in many ways to the work presented in this thesis, and they should not go unmentioned.

First and foremost, I have to thank my supervisor, Prof. Dr. Reinhard Klein, who was an inexhaustible source of inspiration and guidance. With his enthusiasm for computer graphics and related fields he created in only a couple of years a working environment here at the university of Bonn as creative and fruitful as it is. For invaluable feedback I owe thanks also to Prof. Dr. Andreas Weber who was always available on short notice for advice in so many things including (but not restricted to) computer graphics, photography, and travelling.

My sincere appreciation is due to the various co-authors of the papers I published over the years, and I am also sincerely indebted to my current and former colleagues here in the computer graphics group in Bonn, especially Marcin Novotni, Michael Guthe, Ruwen Schnabel, Patrick Degener, and Simone von Neffe. It is obvious to me that my research was only possible in the collaborative and friendly atmosphere that they provided. Particular thanks go to my office mates Ralf Sarlette and Mirko Sattler; the exciting and comprehensive discussions on matters of graphics kind, politics, and the countless patent-pending ideas on how to get rich will not be forgotten. Likewise I owe thanks to Gero Müller and Jan Meseth for compassionately discussing the terrible defeats or glorious victories of our favourite football teams.

It is with deep gratefulness that I think of my family, without whose love and support this work would not have been possible. I would have loved my father, Heinrich Bendels, to live and see this thesis of mine.

Part I

Introduction

Chapter 1

3D Geometry in the Cultural Heritage Domain

Making pieces of art or of particular historic importance available to an audience as wide as possible is a key interest of historians, archaeologists and museums' curators. Ideally, from a didactic point of view, each artefact would be demonstrated within its historic and semantic context, maybe even giving the observer the opportunity to interact and participate, granting access to everyone interested. Aside from the inherent and unsolvable dimensionality problem that any object can be demonstrated physically in one state representing a certain temporal snapshot only – even though it may have underwent important changes over the course of time, these requirements conflict with another fundamental interest at the very heart of every historian: The *preservation* of the objects under his auspices.

Generation of digital three-dimensional copies of the historic artefacts can build a bridge between these discordant and contradictive requirements associated with the handling of valuable artefacts. It therefore comes to no surprise that 3D content generation has gained (and is gaining at an increasing speed) much attention from the cultural heritage community, even although 3D photography is still a sophisticated process that comes yet nowhere near the practicability and ease of use of traditional 2D photography. Nevertheless, the use of computer graphics methods opens up manifold opportunities and paves the way to novel solutions to long standing challenges associated with standard tasks in the cultural heritage domain.

Among the most imminent concerns of museums curators is the documentation and cataloguing of the vast amount of artefacts in the museums' inventories. Not infrequently do in particular high ranked museums have a backlog of up to 50%, and only a fraction of the valuable artefacts in a museum's possession can typically be displayed due to limited space and limited personnel for preparation and handling. Instant and random access to the full spectrum of the items hidden in storage rooms and depots is not even envisioned yet for many museums. Here, 3D photography has the potential to tackle traditional challenges and facilitate novel ways for dissemination and accessibility in a manner unknown with traditional photography.

In addition to this, the use of computer graphics methodologies in the cultural heritage domain can be expected to boost (and is doing so today) a number of applications that are feasible for the first time with the aid of captured 3D geometry:

- Quantitative / statistical analysis
- Restoration planning / documentation
- VR / AR applications
  - Virtual reassembly
  - virtual stress tests / check for plausibility of previous reconstructions
  - Virtual historic/spatial contexts
  - Virtual reconstruction in different states of evolvement
- Prototype generation for
  - Manufacturing of restoration parts
  - Mold generation for replica generation or merchandising
  - Mold generation for packaging and transport
  - Precisely formed supports / stands.

Although for most of the aforementioned applications a faithful digital reproduction of the object under consideration is required and suffices, the endeavour to exploit the acquired data in education and entertainment applications also calls for methodologies to perform artistic and creative operations. To successfully enable experts in the field to incorporate their knowledge into the virtual restoration, make-up and presentation, however, these methodologies have to fulfill some specific requirements:

- The modelling interface must be as simple and intuitive as possible, as future users can be expected to be experts in a field different from computer science
- Precise definition of the region of influence of an editing operation is obligatory, as in particular historians always need to be able to distinct modified from original data.
- The modelling technology cannot be restricted to the *deformation* of the object only, it is essential that also considerable material may be added or removed, including parts of other objects.
- Although detail preservation in this context is a virtue, it stands back behind the importance of the possibility to reconstruct or recreate fine surface detail in regions of added material to match the surrounding of the editing region.

CHAPTER 2

Basics on 3D Geometry Processing

Understood as a general scientific term, *Modelling* refers to the creation of a (typically simplified) representation of a system or phenomenon that retains specifically those properties of the original required by the application or scientific question at hand (cf. [Costello 1991]). In a less abstract sense, a *model* can also be an artificial instance of a physical object which is stripped from all its properties that are irrelevant in the respective context. As such, the prototypical clay miniature that designers create in the early stages of conceptualising a new car can serve as an example of a model, in this case capturing *shape* as the most important aspect. In the present thesis it is this last type that is referred to as models – virtual, digital descriptions of a physical object. In principle, such models can be generated in two fashions, either by *ab initio*-creation or by capturing the desired properties using

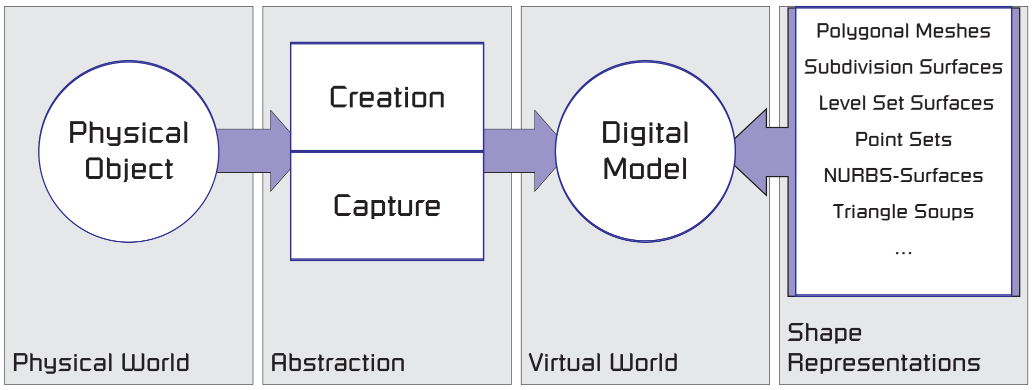


Figure 2.1: *Model generation / Abstraction.*

some kind of 3D photography, e.g. using Laser range scanners, structured light, tactile sensors, or volumetric methods like CT or μCT. The properties that are primarily required to be captured in the model are *shape, appearance*, and sometimes some aspects of the *material* the physical object is made of.

With respect to shape, a number of representations have been developed over the years in computer graphics and can be considered mainstream: Polygonal meshes, subdivision surfaces, level set surfaces, point sets, NURBS-surfaces, to name a few. All of these have their specific strengths and weaknesses, depending on the application they are used in – not all representations are equally suitable for all applications for all types of models, just as not all processing strategies are equally suitable for all representations.

Unlike in the automotive and manufacturing industries, where CAD-tools dominate the digital creation process, creating 3D models from scratch is generally not the method of choice in the cultural heritage sector, as it would antagonise many of the applications mentioned above, in particular with respect to documentation and analysis. Various projects do exist that aim at generating virtual models of buildings and architecture using CAAD-tools, and the grammar-based modelling of historic architecture is currently gaining increasing research attention (see [Havemann 2005]), but modelling in particular sculptures etc. is yet virtually unfeasible. The object representations predominantly used in this thesis are therefore those directly related to the data capturing process, i.e. point sets, implicit representations as distance fields, and triangle meshes. The following sections will give a short overview over some of the most popular surface representations.

2.1 Digital Object Representations

2.1.1 Point Sets

Among the more popular representations of a 2D-surface in 3D, the point set representation certainly constitutes the conceptually simplest. For a given 2-manifold surface $\mathcal{S}$ in $\mathbb{R}^3$, it consists of no

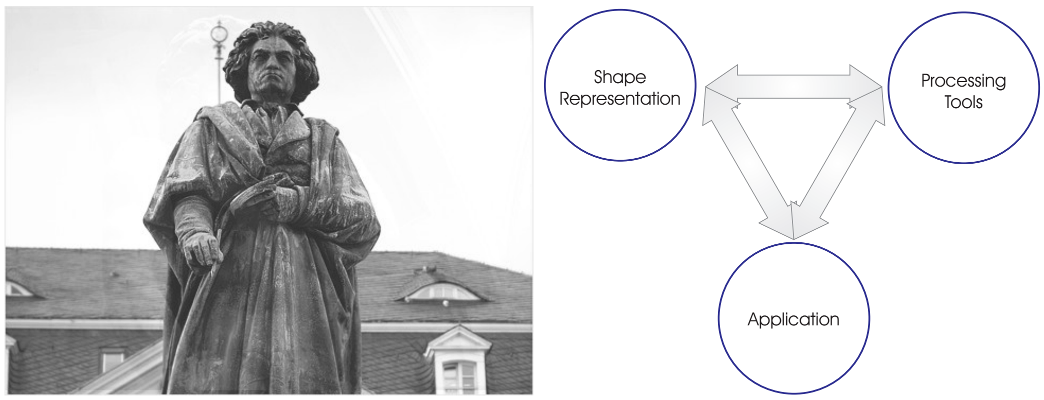


Figure 2.2: *Shape, its digital representation, the processing tools, and the application have a natural influence onto each other.*

more than a finite set $\mathcal{P}$ of point samples of $\mathcal{S}$:

$$\mathcal{P} = \left\{ \mathbf{p}_1, \ldots, \mathbf{p}_N \in \mathbb{R}^3 \right\},$$

where $N \in \mathbb{N}$, and $\mathbf{p}_i \in \mathcal{S}$ for all $i = 1, \ldots, N$. In its basic formulation no additional knowledge such as connectivity, spatial structure, etc. is required. Nevertheless, a point's position is often paired with other attributes like colour and normal, and the resulting n-tuple is usually referred to as *surfel* ([Pfister et al. 2000]). For many applications, e.g. rendering and the transformation into other surface representations (see section 2.4), in particular the *surface normals* are required and either captured together with $\mathcal{P}$ from the surface $\mathcal{S}$ or derived from $\mathcal{P}$ using local surface analysis operators. Throughout this thesis, $\mathbf{n}(\mathbf{p})$ denotes the surface normal at some point $\mathbf{p} \in \mathbb{R}^3$.

Representing surfaces through point sets has become increasingly popular in the past few years. One reason for this is the spreading of affordable 3D capture technology in form of laser range scanners outputting easily millions of point samples, and thereby producing a faithful sampling of the scanned surface. The other reason, that is equally important, is the development of powerful algorithms that prepare the ground for numerous applications to be employed directly on the point set itself without the need to perform a full-scale reconstruction first.

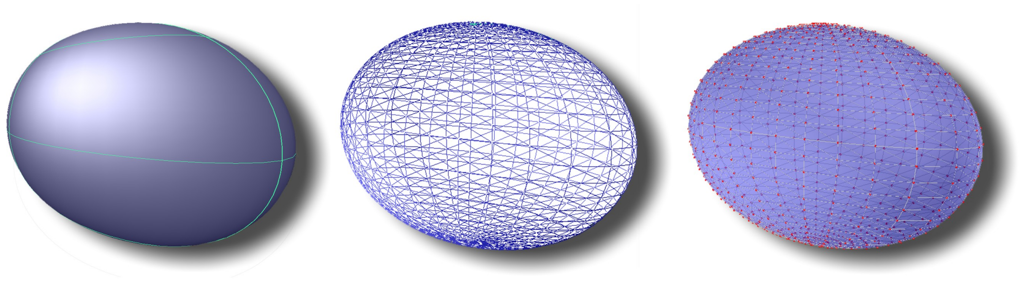


Figure 2.3: *An ellipsoidal surface represented (from left to right) implicitly as level set* $\{\mathbf{p} \in \mathbb{R}^3 \mid \mathbf{p}_x^2 + 0.5\mathbf{p}_y^2 + \mathbf{p}_z^2 - 1 = 0\}$*, as triangle mesh, and as point set (overlaid over the triangle mesh for illustration purposes).*

2.1.2 Parameterised Surfaces

Let $I \subseteq \mathbb{R}$ be an interval, and let $f : I \times I \to \mathbb{R}^3$ be a continuous function. Then a parameterised surface $\mathcal{S}$ is defined as the set

$$\mathcal{S} = \left\{\mathbf{p} \in \mathbb{R}^3 \mid \exists (u, v) \in I \times I : \mathbf{p} = f(u, v)\right\}.$$

An example of this representation are the famous *Non-uniform Rational B-Spline* (NURBS) surfaces (see e.g. [Farin 1990]) that were used extensively (and are still today) in the engineering industries to describe the building components of cars. However, even in the automotive industries, where the desirable computational properties of NURBS surfaces are indispensable, their use is currently pushing limits, as the number of trimmed NURBS patches for a single car reach the order of millions. Constructing parameterised surfaces for captured physical models is non-trivial and computationally prohibitive.

2.1.3 Triangle Meshes

One special case of parameterised surfaces are triangle meshes. A triangle mesh is a piecewise linear surface that can be formulated as the pair $(\mathcal{V}, \mathcal{E})$, consisting of a set $\mathcal{V}$ of *vertices* in $\mathbb{R}^3$, representing the surface's geometry and a set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ of *edges* capturing the connectivity and the topology of the surface. The edges in $\mathcal{E}$ form a planar graph whose faces are triangles in $\mathbb{R}^3$.

One reason for the popularity of this type of surface representation is the fact that nowadays' graphics hardware is heavily tuned to rapidly handle large amounts of triangle mesh data, such that more than 100 million triangles can be rendered per second (according to NVIDIA, up to 181 million triangles per second on the Quadro FX 4500).

2.1.4 Implicit Representations

In contrast to parameterised surfaces which can be considered the *image* of a function, level set surfaces are defined implicitly as the *kernel* of a functional: Let $F : \mathbb{R}^3 \to \mathbb{R}$ be a continuous functional, then a level set surface $\mathcal{S}$ is defined as the set

$$\mathcal{S} = \left\{ \mathbf{p} \in \mathbb{R}^3 \mid F(x,y,z) = 0 \right\}.$$

Algebraic surfaces, e.g. the ellipsoid $\{\mathbf{p} \in \mathbb{R}^3 \mid \mathbf{p}_x^2 + 0.5\mathbf{p}_y^2 + \mathbf{p}_z^2 - 1 = 0\}$, depicted in figure 2.3, are examples of this type of surface representation, where the surface points are roots of polynomials up to a given degree. Another, more abstract approach to implicit surfaces, is the zero-set of the distance field defined by the surface itself:

$$F(\mathbf{p}) = \delta(\mathbf{p}, \mathcal{S}),$$

where δ is some appropriate (signed) distance measure between a point in $\mathbb{R}^3$ and the surface (or some surface approximation, see section 2.4).

By definition, implicit surfaces deliver an ubiquitous availability of inside/outside information by simply evaluating the respective functional F. In addition to that, implicit surfaces are powerful representations of topologically complex surfaces, that can easily handle topological changes during modelling. On the other hand, implicit surfaces do not enable any direct access to the surface itself. For an elaborate coverage of implicit surfaces see [Bloomenthal 1997].

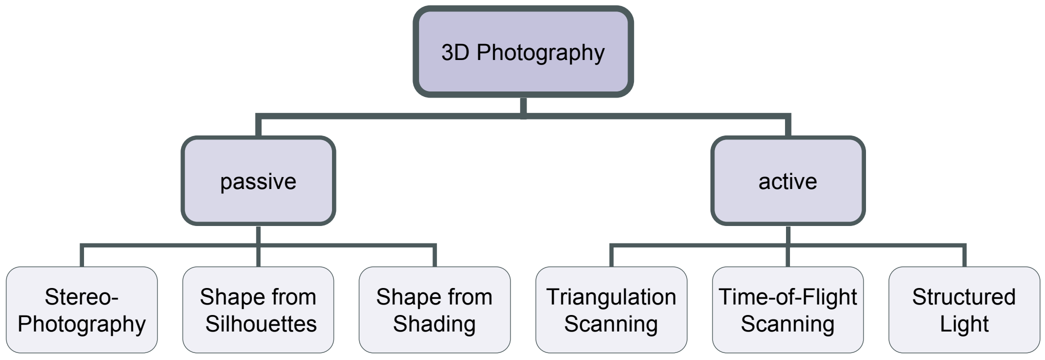


Figure 2.4: *3D data acquisition taxonomy.*

2.1.5 Other Representations

All of the aforementioned surface representations are concerned with the geometry of the surface to be represented. In some applications and in particular for rendering, however, its *appearance* is more important. As a consequence, *image-based methods* that allow regarding the object from arbitrary view-points and under arbitrary lighting conditions have been developed. It has been demonstrated e.g. by Hawkins et al. [2001], that for demonstration and illustration purposes of an object itself, it can be represented by a dense set of photographic images without explicit geometry reconstruction. By basing the representation on photographs only, this approach is viable even for objects for which a faithful geometry capture is at least challenging, e.g. materials such as leather, feather, or fur. Nevertheless, numerous applications involve at least coarse geometry. Chapter 5 will discuss this hybrid type of representation where geometric fine detail is represented by images whereas the coarse geometry is represented explicitly as triangle mesh.

2.2 Acquisition

As of today, the technologies available to digitise three-dimensional physical objects can typically be categorised into three groups: *Laser-based*, *Computer Tomography* (CT or μCT), and *Light Field* methods that are based on the exploitation of dense sets of photographic images.

	Surface	Volume
Contact	Tactile Sensors	Histologies
Non-Contact	3D Photography	CT, μCT, MRT

Among the methods that use Laser-emitting devices, one can in turn distinguish three different range finding principles: Time-of-Flight, triangulation, and interferometric scanners. Due to the limited precision of time-of-flight scanners and the considerable cost of interferometric scanners, triangulation scanners are most widespread. They combine a simple setup (compared to interferometric scanners) resulting in mid-price devices with a reasonable precision of few microns under ideal conditions. For a given viewpoint, these non-contact capturing devices record a rectangular array of depth values (a so-called *Range Image*) by tracking contour lines projected onto the physical object (cf. figure 2.5). The depth values represent the distance between the capturing camera and the corresponding point on the object's surface and therefore encode the point's position in space relative to the scanning device. As a precise tracking of the global position of the scanner is generally unfeasible, each range image is given in its own coordinate system. This necessitates the transfer of each individual range image into one common, global coordinate system – a process that is usually referred to as *registration*. This next step in the acquisition pipeline after capturing the raw data will be described in the following section (section 2.3).

Computer tomography methods raster the space containing the object into a finite set of voxels and record density values for each voxel. Here, the full data is recorded in a common coordinate system such that no registration is necessary. Moreover, computer tomography allows *a look on the inside* of the object, which is particularly useful for objects with an intricate interior structure. On the other hand, the fact that only density values are recorded necessitates contour extraction algorithms to be applied before an actual surface can be generated. This step of the acquisition

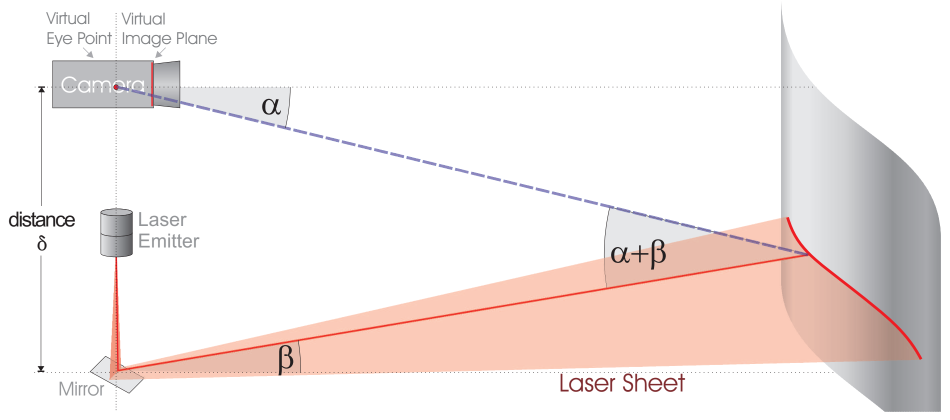


Figure 2.5: *Triangulation scanning principle: A horizontal line is projected onto the object and recorded by an attached camera. For known laser and view directions, the enclosed angle $\alpha + \beta$ allows precise depth computation.*

pipeline is usually referred to as *reconstruction* and will be the topic of section 2.4.

Data acquisition using CT suffers from a limited resolution, while μCT delivers high resolution recordings at the cost of considerably confined measurement dimensions.

Acquiring the light field representation of an object involves the sampling of the space of all light directions and of all viewing directions. To this end, the object is lit from predefined lighting positions and under each such elementary illumination, a photograph is taken from equally predefined viewing positions. The acquisition setups presented so far ([Malzbender et al. 2001], [Hawkins et al. 2001], [Hawkins et al. 2005], [Koch 2006]) differ mainly in the sampling density with respect to light and viewing directions, and the extent to which the acquisition is parallelised. In particular the setup developed by Sarlette et al. described in [2006] delivers fast acquisition times of less than 20 minutes (three times as much for high dynamic range recordings) by massive parallelisation.

2.3 Registration

Range images captured with a laser range scanner are point clouds, each sampling a certain part of the digitised object and

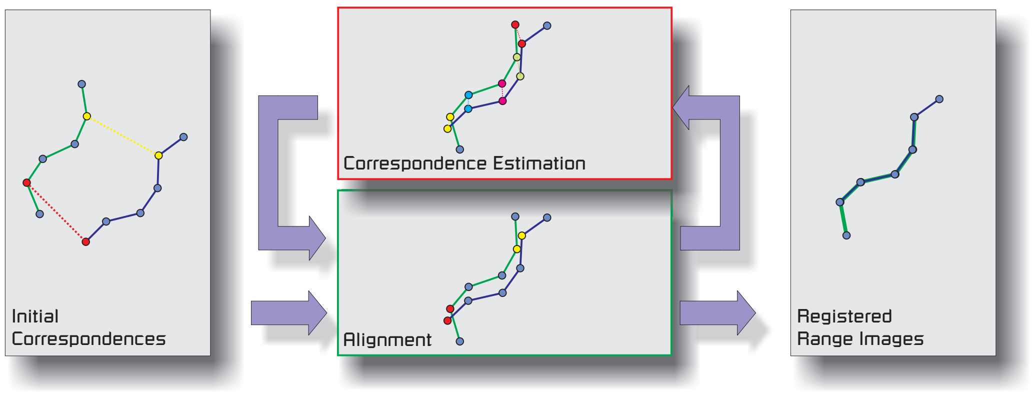


Figure 2.6: *Iterative Closest Points process flow.*

given in its own local coordinate system. To derive a complete closed surface sampling, the range images have therefore to be transferred into a common global coordinate system. This registration step is traditionally performed by a manual pre-registration step that coarsely pairwise aligns the range images, followed by an iterative automatic fine alignment.

The goal of range image alignment is to find a coordinate transformation that maps points representing an identical position on the scanned surface to coinciding positions in space. Pairs of such points in the overlapping regions of two range images are called *corresponding*. Unfortunately, the correspondences between recorded range images are generally unknown and – although at least in part easily identifiable for a human observer – not accessible for an automated registration. It is worth noting that knowledge of the complete set of correspondences is equivalent to knowing the coordinate transformation between two range images. The concept of most successful approaches to (semi-)automatic registration is therefore an iterative procedure that alternately estimates correspondences and transformations (cf. figure 2.6).

Iterative Closest Points

Let $\mathcal{P} = \{\mathbf{p}_1, \dots, \mathbf{p}_N\} \subset \mathbb{R}^3$ be a range image, and let $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\} \subset \mathbb{R}^3$ be a point set with which $\mathcal{P}$ is to be aligned. In the context of registration, $\mathcal{P}$ is also denoted as the *data*, while

$\mathcal{X}$ is called the *model*. Let $\mathcal{C}_\mathcal{X} : \mathbb{R}^3 \to \mathcal{X}$ be the operator that finds for any point $\mathbf{p}$ in the Euclidean space the closest point $\mathbf{x_p}$ in the model $\mathcal{X}$:

$$\mathbf{x_p} = \mathcal{C}_\mathcal{X}(\mathbf{p}) = \arg\min_{\mathbf{x}\in\mathcal{X}} \; d(\mathbf{p}, \mathbf{x})$$

according to some distance function d, typically the Euclidean distance

$$d(\mathbf{p}, \mathbf{x}) = \langle \mathbf{p} - \mathbf{x}, \mathbf{p} - \mathbf{x} \rangle^{1/2}.$$

Here, $\langle ., . \rangle$ denotes the standard scalar product in $\mathbb{R}^3$.

The key component of the popular ICP method introduced by Besl et al. [1992] is to assume that the **geometrical proximity** of $\mathbf{p}$ to $\mathcal{C}_\mathcal{X}(\mathbf{p})$ indicates correspondence, i.e. **semantic equivalence**. ICP consequently interprets $\{(\mathbf{p}_i, \mathbf{x}_{\mathbf{p}_i})\}_{i=1,\dots,k} \subset \mathcal{P} \times \mathcal{X}$, with $k \leq N$, to be corresponding point pairs and tries to find a rotation $\mathbf{R}$ and a translation $\mathbf{t}$, such that

$$\frac{1}{k} \sum_{i=1}^{k} d(\mathbf{x}_{\mathbf{p}_i}, \mathbf{R}\mathbf{p}_i + \mathbf{t})^2 \to \min_{\mathbf{R},\mathbf{t}}!$$

After the minimising transformation $(\mathbf{R}, \mathbf{t})$ is found, it is applied to $\mathcal{P}$ and the procedure is iterated, each time updating the correspondences using the closest-point-operator $\mathcal{C}_\mathcal{X}$.

Initially, the above assumption is unjustified. The minimisation is therefore prone to lead to a local minimum that is arbitrary distant to the true solution. To counteract this, the iteration is initiated by a manual identification of corresponding point pairs, i.e. by selecting and declaring a sparse set of corresponding point pairs in the data and the model point set.

Although several extensions to the ICP algorithm have been developed over the years that help avoiding local minima and circumvent false correspondence computations, the fully automatic registration without manual pre-alignment is still a matter of current research. Chapter 4 will give a more detailed overview over the existing approaches and will also introduce a solution to this problem that exploits 2D-intensity images that are frequently recorded during scanning by most available off-the-shelf Laser range scanners.

2.4 Reconstruction

Given powerful and efficient algorithms to directly handle large point sets outputted from the previous stages in the content creation pipeline, many applications can deal with the point set representation itself and no further processing is required. For other applications however, a closed surface representation in form of a polyhedral surface (such as a triangle mesh) is still required or can at least be exploited for performance or efficiency reasons.

The goal of surface reconstruction as formulated by Hoppe et al. [1992] is therefore as follows:

> Given a surface sampling $\mathcal{P}$ of points $\mathbf{p}_i \in \mathbb{R}^3$, $i = 1, \ldots, N$ on or near a surface $\mathcal{S}$, determine a surface approximation $\hat{\mathcal{S}}$ to $\mathcal{S}$.

This conversion from a point sampling into a surface approximation $\hat{\mathcal{S}}$ is typically performed as a two-step process: Firstly, an implicit representation is constructed from the point set – for instance in form of the zero set of the surface's distance field ([Hoppe et al. 1992],[Ohtake et al. 2003]), of a radial basis function fitted to the given surface data ([Carr et al. 2001]), or in form of the stationary set of a projection operator ([Levin 2003]). In a subsequent *contouring* step, this implicit representation is converted into a polygonal mesh by an iso-surface extraction algorithm.

This two-step reconstruction paradigm is also applied if faced with data from intermediate stages in industrial CAD processes which often suffers from topological inconsistencies, cracks in the tessellation etc. In this case the implicit representation is chosen to faithfully reproduce the surface properties, whereas the contour extraction can be tuned to generate surface meshes with the desired properties.

2.4.1 Moving Least Squares

The basic idea of the well-known moving least squares surface interpolation scheme by Levin [2003] is to define $\hat{\mathcal{S}}$ as stationary

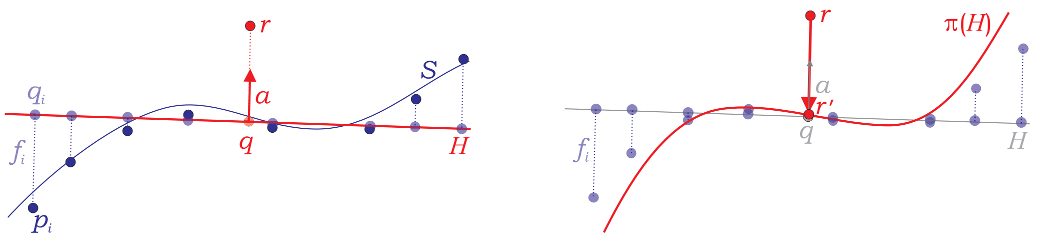


Figure 2.7: *The moving least squares projection operator. Left: For a given point $r \in \mathbb{R}^3$ near the approximated surface $\mathcal{S}$, the minimising hyperplane H is found and a local coordinate system with origin at q is defined. Right: The projection operator is defined as a mapping of r onto the best fitting polynomial $\pi \in \Pi^2_m(H)$.*

set of a projection operator P_m that projects points close to sufficiently sampled regions of the original surface $\mathcal{S}$ onto $\hat{\mathcal{S}}$: Let r be a point near the approximated surface $\mathcal{S}$; a hyperplane $H = (a, D)$ with normal $a \in \mathbb{R}^3$, $\|a\| = 1$ and distance $D \in \mathbb{R}^{\geq 0}$ to r is defined s.t.

$$\sum_{\mathbf{p}_i \in \mathcal{P}} \left(\langle a, \mathbf{p}_i\rangle - D\right)^2 \theta(\|\mathbf{p}_i - q\|) \rightarrow \min_{a,D}!, \tag{2.1}$$

where θ is a non-negative weight function and q is r's projection onto H (see figure 2.7, left). The resulting hyperplane H defines a local reference domain with its origin located at $q \in H$.

In a second step, a polynomial $\pi \in \Pi^2_m(H)$ of degree m is fitted to the residuals $f_i = \|\mathbf{p}_i - q_i\|$ such that

$$\sum_{\mathbf{p}_i \in \mathcal{P}} \left(\pi(q_i) - f_i\right)^2 \theta(\|\mathbf{p}_i - q\|) \rightarrow \min_{\pi}!$$

becomes minimal. It is worth noting that in the above equations the weight function is evaluated depending on the distance between the sample points $\mathbf{p}_i$ and the origin q, i.e. the projection of r onto H.

The MLS surface finally is defined to be the stationary set of the projection operator that maps a point r near S to the corresponding point on the polynomial:

$$r \;\mapsto\; r' = q + \pi(0)a.$$

In many cases, normals are not known by measurement and need to be assigned to each point $\mathbf{p} \in \mathcal{P}$, and it is tempting to use the normal $a(\mathbf{p})$ of the approximating hyperplane H as defined above. However, as stressed by Alexa and Adamson [2004], $a(\mathbf{p})$ is not necessarily collinear with the surface normal, although it is frequently employed as such. For further details cf. the above publication, as well as [Adamson & Alexa 2003] and [Adamson & Alexa 2004].

As pointed out by Klein and Zachmann [2004], the weight function θ in equation (2.1) depends on the Euclidean distance, which does not respect any topology potentially present in the data, and hence may declare points "close" that are indeed, at least topologically, far away. Although on first sight this corresponds well to the fact that point sets do not explicitly store topology information, Klein and Zachmann correctly argue that proximity graphs can be used to approximate geodesic (and therefore surface-inherent) distances to overcome these limitations at the cost of only little storage overhead.

Although moving least squares surfaces in their original formulation are smooth by definition and therefore unable to reproduce sharp features, recent approaches have introduced *piecewise* smooth surface representations based on the MLS. See e.g. Fleishman et al. [2005], who used statistics methods to detect and preserve sharp features present in point sampled data in the MLS representation.

2.4.2 Radial Basis Functions

In contrast to the local approximation nature of the moving least squares scheme, where the approximating function (the *approximant*) is a low degree polynomial defined over a local domain available only in close vicinity to the approximated surface (the *approximand*), *radial basis function (RBF)* interpolation derives one implicit function whose zero set globally defines the approximating surface $\hat{S}$.[1] Its task is therefore to find a function $\hat{s}$ such

[1]In order to avoid confusion between interpolation and approximation, please note that the approximation $\hat{\mathcal{S}}$ of the original surface $\mathcal{S}$ is derived in this section via interpolation

that

$$\hat{s}(\mathbf{p}) = 0 \text{ for all } \mathbf{p} \in \mathcal{P}.$$

In order to avoid trivial solutions like $s = 0$, boundary constraints are inserted that define non-zero values for off-surface points, one obvious choice being the signed distance of these points to the approximated surface (cf. [Carr et al. 2001]). The complete interpolation problem can then be stated as:

> Given a surface sampling $\mathcal{P}$ of points $\mathbf{p}_i \in \mathbb{R}^3$, $i = 1, \ldots, N$ on or near a surface $\mathcal{S}$, and further a set of off-surface points $p_j \in \mathbb{R}^3$, $j = N+1, \ldots, N+M$ find a function $\hat{s}$ such that
>
> $$\hat{s}(\mathbf{p}_i) = s_i \text{ for all } i = 1, \ldots, N+M$$
>
> where $s_i = 0$ for $i = 1, \ldots, N$. For $i = N+1, \ldots, N+M$, s_i denotes the off-surface points' distance to the approximated surface.

In the above problem statement, the space from which the optimal function $\hat{s}$ may stem is unspecified. In practice, the choice of an appropriate function space is influenced by additional smoothness assumptions for the approximand, which lead to optimality criteria for the approximant, and by computational considerations, which typically lead to finite dimensional function spaces.

Both conditions are fulfilled by setting the space of allowable interpolants to be

$$\left\{ s(x) = p(x) + \sum_{i=1}^{N+M} \lambda_i \, \phi(\|x - \mathbf{p}_i\|) \right\},$$

where $\phi : \mathbb{R}^{\geq 0} \to \mathbb{R}$ are the so-called radial basis functions and $p(x)$ is a low degree polynomial.

The simplest example for an RBF interpolation is the interpolation with finite linear combinations of translations of the radially symmetric function $\phi(r) = r$:

$$s(x) = p(x) + \sum \lambda_i \|x - \mathbf{p}_i\|,$$

of the sample points in $\mathcal{P}$. In this sense, $\hat{\mathcal{S}}$ is an approximant to $\mathcal{S}$ and an interpolant to $\mathcal{P}$.

with a linear polynomial p and the Euclidean norm $\|\cdot\|$. It was shown by Duchon [1977] that the *smoothest* interpolant in the space of Beppo-Levi distributions on $\mathbb{R}^3$ is guaranteed to have this particular form. (See e.g. [Carr et al. 2001] for further details.)

Other examples for frequently employed radial basis functions include

$\phi(r) = r$	*Biharmonic*
$\phi(r) = r^2 \log(r)$	*Thin plate*
$\phi(r) = r^3$	*Triharmonic*
$\phi(r) = e^{-\alpha r^2}$	*Gaussian*

The specific choice of a radial basis function for a given interpolation problem strongly influences not only the resulting interpolant but also the computational effort to solve for the required coefficients λ_i and those of p. In particular, radial basis functions with global support (for instance the triharmonic RBF $\phi(r) = r^3$) deliver fair surface approximations that are able even to cover large holes in the input sampling at the price of a dense matrix in the corresponding linear system (see below). RBFs with a more local support lead to sparse matrices that can be solved far more efficiently, but may generate surfaces with undesired properties.

Let $\pi_1, \dots, \pi_l$ be the basis for the space of polynomials up to degree l and let $c_1, \dots, c_l$ be the corresponding coefficients of p in this basis. The linear system for the desired interpolant can then be written as

$$\begin{pmatrix} \Phi & \Pi \\ \Pi^T & 0 \end{pmatrix} \begin{pmatrix} \lambda \\ c \end{pmatrix} = \begin{pmatrix} s \\ 0 \end{pmatrix},$$

where $\Phi_{ij} = \phi(\|\mathbf{p}_i - \mathbf{p}_j\|)$ and $\Pi_{ij} = \pi_j(\mathbf{p}_i)$. This linear system also includes the orthogonality conditions that require

$$\sum_{i=1}^{N} \lambda_i = \sum_{i=1}^{N} \lambda_i \mathbf{p}_i = 0.$$

Regarding this linear system, it becomes obvious why RBF interpolation has traditionally been considered inappropriate for reconstruction purposes where point clouds easily reach the size of millions of points. Even for reduced problem sizes in the context of surface modelling, solving the above linear equation remains computationally involved, although recent approaches have proven their feasibility even in real-time applications [Botsch & Kobbelt 2005].

The biggest advantage of RBF interpolation techniques is that they are able to derive smooth interpolants under only very mild conditions on the placement of the RBF centres – by construction, since the placement of the centres itself influences the function space. For function spaces that are defined without respect to the positions of the points to be interpolated, it is typically not difficult to construct examples that lead to singularities and non-invertible matrix representations.

For in-depth reading please refer to [Buhmann 2003], [Duchon 1977], [Carr et al. 2001] and [2003], and the references given therein.

2.4.3 Multi-level Partition of Unity Implicits

One of the most efficient approaches known to date to build a surface representation from large sets of sampled surface points are the so-called *Multi-level Partition of Unity Implicits* introduced by Ohtake et al. [2003]. This approach derives an implicit surface representation from point samples by computing local quadratic surface approximations in octree cells (the so-called *local shape functions*[2]). The local surface approximations are then blended in order to generate a closed, smooth surface. The name MPU is due to the fact that the blending functions (the weights) sum to one at every input point.

More specifically, let $\mathcal{P}$ as usual denote a set $\{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$ of sample points representing a 2D surface $\mathcal{S}$ in $\mathbb{R}^3$ with associated surface normals $\{\mathbf{n}_1, \ldots, \mathbf{n}_N\}$. The goal is now to derive, in an adaptive manner a function $f : \mathbb{R}^3 \to \mathbb{R}$ whose zero level set approximates the unknown underlying surface.

[2]The name *local shape functions* is adopted here for reference reasons. This should not cause any confusion, however, with the notion of shape functions for modelling used in chapter 8.

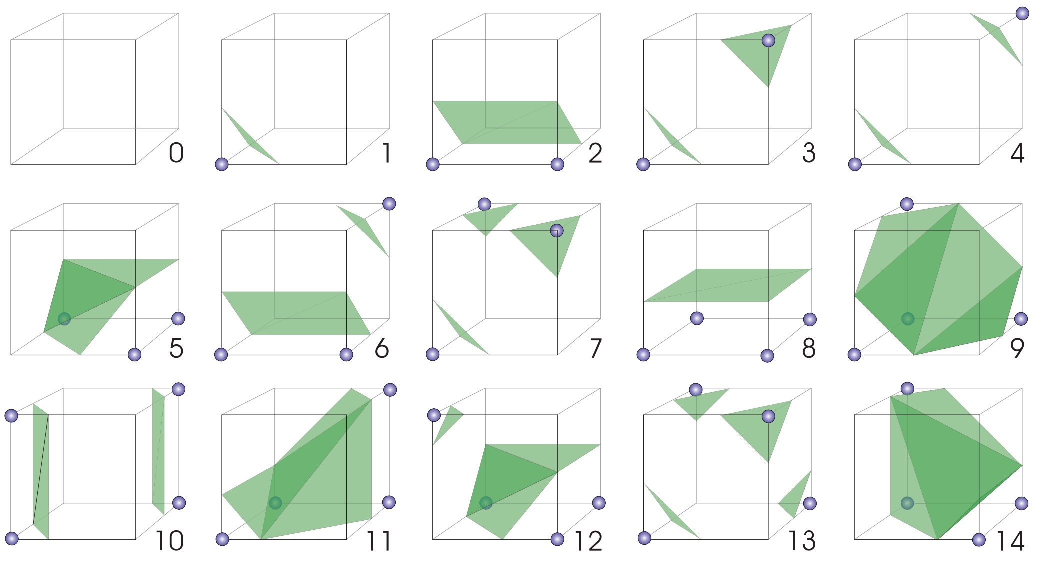


Figure 2.8: *Marching Cubes configurations.*

To this end, the space surrounding the point set is subdivided adaptively and local surface approximations are constructed hierarchically in a top-down fashion. For each cell Ω of the space subdivision scheme, a quadratic functional is fitted. It is then checked, if the approximation error of this function with respect to the points in Ω exceeds a certain threshold. In this case, Ω is subdivided and the procedure is performed recursively on its child cells.

During fitting, the points contained in Ω are analysed to select the specific type of functionals to be used for approximation. In order to be able to also represent edge- and corner-shaped sharp features, the options include a heuristic to detect sharp features and the use of *piecewise* quadratic functions.

The main advantages of the MPU as implicit surface approximation is that it is error-adaptive and fairly fast.

2.4.4 Contour Extraction

The standard approach for contour extraction from implicit functions is the famous *Marching Cubes*-algorithm introduced by Lorensen and Cline [1987]. In its original formulation, it generates a piecewise linear approximation of the implicit surface based only on inside / outside information that is specified at the nodes of a uniform grid. Exploiting symmetry relations, the set of possible inside / outside combinations at the eight grid nodes belonging to a grid cell can be reduced to 15 (see figure 2.8). Each grid cell is then processed and triangulated individually. If a consistent strategy is pursued to resolve ambiguous cases, the marching cubes algorithm is guaranteed to produce a closed 2-manifold mesh that separates nodes marked inside from those marked outside.

Over the years, numerous extensions and improvements of the marching cubes algorithm have been introduced, among others enabling the reconstruction of features inside octree cells ([Kobbelt et al. 2001]), and allowing for adaptive grids to be handled ([Bloomenthal 1988],[Shu et al. 1995]). Adaptivity was restricted by the fact, though, that adjacent grid cells were not allowed to differ by more than one octree level. Combining the dual surface nets approach by S.F.F.Gibson [1998] with ideas from [Kobbelt et al. 2001], this restriction was relieved by the dual contouring approach introduced by Ju et al. [2002]. Later, the dual surface nets approach was adapted to kd-trees by Greß and Klein [2003; 2004], further enhancing adaptivity and allowing even for thin solid structures to be faithfully reconstructed. For further reading, cf. [Greß & Klein 2004] and the references therein.

CHAPTER 3

Contributions

Motivated by the numerous and manifold applications in the cultural heritage sector that become viable – in some cases even for the first time – using computer graphics techniques, this thesis investigates methods for 3D geometry processing under consideration of the specific aspects of its use in this particular field. The work presented in this thesis summarises (and extends) work published in various papers as listed in section 9.7.
The main contributions are:

- A selection of 3D model generation algorithms that enable users to efficiently produce digital copies of existing models in a fully- or semi-automatic fashion on the basis of various types of data sources (part II). Of particular relevance for the subsequent contributions is a fully automatic range image registration approach that exploits features contained in the 2D photographic images typically recorded together with the range images

- A fully automatic hole detection approach for point sampled surfaces that paves the way for a context-based, hierarchical surface completion algorithm for point sampled surfaces that is able to reconstruct large scale as well as fine detail features in the hole region (part III)

- An interactive and semi-automatic modelling paradigm (part IV) that allows for intuitive and efficient free-form modification of 3D surface data and can seamlessly be integrated into the surface completion previously introduced in part III.

The methods and algorithms presented in this thesis are described and motivated with their applicability in the context of cultural heritage that offers a variety of applications and at the same time poses some specific demands. Nevertheless, most of the contributions of this thesis are applicable and relevant as such in many other contexts.

The thesis is organised in four parts. After the introductory part, part II (from page 27) deals with the generation of 3D models based on range images and from dense image sets. Focussing on the results from the registered point-based objects, the topic of part III (from page 71) is the repair and completion of the generated models, while the last part (from page 121) of this thesis considers modelling methods, that can also be used in conjunction with the automatic surface completion approach described in the preceding part.

Part II

Model Generation

CHAPTER 4

Reconstructing Geometry from Scan Data

Due to its accuracy, inexpensiveness, and non-intrusiveness, digitising 3D-Objects with Laser-Range Scanners is the method of choice for many applications, ranging from the automotive over the entertainment industries to creative design and cultural heritage applications. However, to produce a complete surface of the object to be digitised, the measurement of a single view seldom provides sufficient data, such that multiple, often dozens of views have to be registered. Registering two views of an object is usually a two-stage process: First, an initial transformation is estimated, which, in turn, is used as a starting point for the second stage, the fine registration.

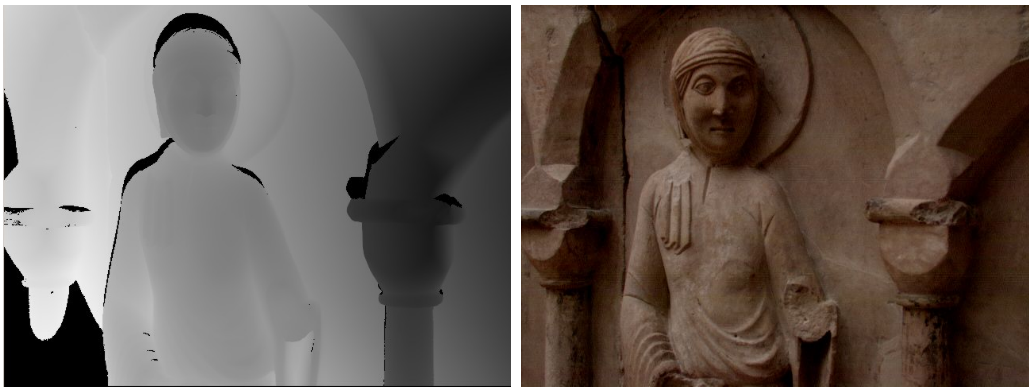


Figure 4.1: *Range and colour image acquired with a laser range scanner (in this case a Minolta Vivid 900)*

The fully automatic registration of multiple range images is still an area of active research in computer graphics. Commercial systems often rely on user-interaction to determine the initial transformation (see [Callieri et al. 2003]), making the pre-registration a tedious and time-consuming task. To overcome this drawback, in some applications additional information available from the scanning process can be exploited to derive the initial transformation: For instance, the relative viewpoint position might be known, e.g. from tracking the scanner position or by using a turntable on which the object to be digitised was situated. Although direct and convenient, this is not always feasible due to the nature of the object, its dimensions or location. Therefore, a common approach is to derive an initial transformation by aligning a small set of corresponding feature points in the range images. These feature points are either found as local geometric features on the surface of the object or by placing additional markers on or in the surrounding of the object. In the former case, robustness of the feature detection is of vital importance, whereas in the latter, special care has to be taken in the placement of the markers [Akca 2003], as markers should be visible from as many viewpoints as possible whilst casting preferably no shadows on the object. Aside from the inconvenience, the placement of markers on the object is *infeasible* in cultural heritage applications, where artefacts to be digitised often must not be touched at all. The need for close-up scans for detailed and spacious objects also prohibits the use of markers placed in the surrounding (see figs 4.1 and 4.12).

On the other hand, scanning devices commonly capture not only geometry but also colour information or light intensities for the scene (cf. figure 4.1). These intensity images are far less subject to noise and as opposed to range images do not exhibit missing values (coloured black in figure 4.1, left). As a consequence, feature points extracted from these images are more robust than those extracted from range images, making them more suitable for correspondence computation. In addition to the robustness, expressiveness and mere number of the features available in the 2D-image information, the key to the ensuing registration steps lies in the fact that the features used here provide *scales* – an indication of how far the surrounding of the feature is also part of the feature. It is this conjuncture of availability, robustness and scale-inherence that allows the formulation of the so-called *feature surface elements* and consequently an efficient automatic high-quality registration.

After solving the pairwise registration procedure, the registration problem has to be solved for the full set of available range images. This becomes necessary as the range scans usually overlap with a number of neighbouring range images. In real-world data sets, the range images will be noisy and erroneous due to *material properties* (colour, shininess, transparency, etc.), *lighting situation*, and *object dimensions* (due to a limited depth of focus in the optical system of the scanner). For each neighbour the bilateral registration will therefore result in more or less differing minimising positions. This non-conformity necessitates mediation among the respective, bilaterally optimal, transformations. In sections 4.2 and 4.4, this problem is solved using a directed cost graph formulation of the multi-view registration task, where the range images constitute the nodes and two nodes are connected by an edge iff the corresponding range images overlap sufficiently. As cost, each edge is attributed with the error induced by registering the two range images corresponding to the adjacent nodes.

The algorithm presented in this chapter is a fully automatic registration approach based on 2D-image feature correspondences which incorporates the following key features:

- No need for special markers
- Robustness with respect to noise and missing geometry data
- Automatic incorporation of additional markers if available

The registration algorithm is incremental in the sense that additional range images can be incorporated into a set of already registered range images very efficiently. The feature detection is performed unilaterally (constant time), whereas the feature matching has to be done with respect to each of the 2D-images in the given set (linear). Finally, the graph relaxation procedure is performed on the full set of range images. Results from previous range image integration can nonetheless be exploited, as extending an already relaxed graph with additional range images converges very fast.

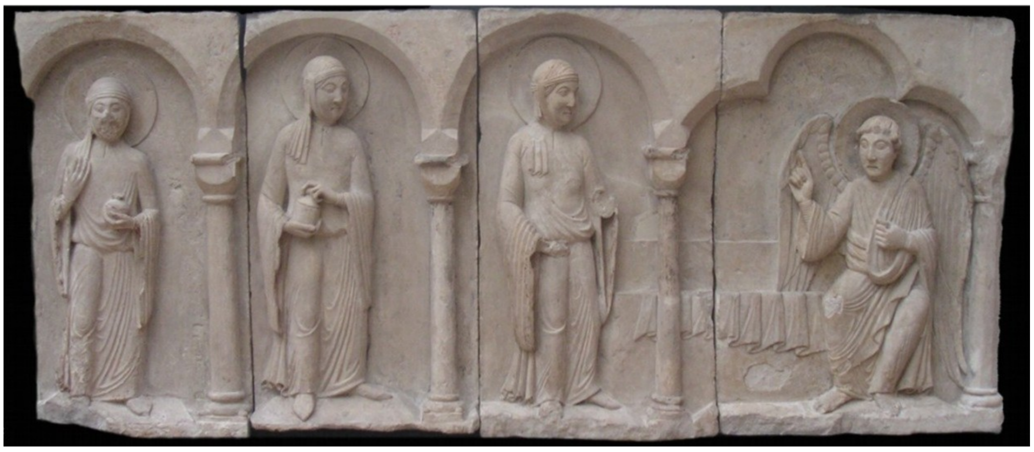


Figure 4.2: *Photograph of a medieval rood screen that was scanned and reconstructed using the approach presented in this paper*

4.1 Related Work

4.1.1 Pairwise Registration

One of the most popular registration methods in literature is the iterative closest pair algorithm (ICP) by Besl and McKay [Besl & McKay 1992] described in section 2.3. It iteratively searches for closest point pairs in two surface patches and optimises the transformation to minimise the distances between these points. However, since this algorithm implicitly assumes that closest points on different patches correspond to each other, it only converges toward a reasonable solution if the patches are roughly pre-aligned. In order to overcome this drawback, various improvements and variants of the original ICP were proposed. This includes verification of closest point pairs by additional attributes like colour or surface normal which is sometimes referred to as the iterative closest compatible point algorithm (ICCP). Furthermore, more sophisticated optimisation schemes were proposed as for example simulated annealing or evolutionary algorithms. [Rodrigues et al. 2002] and [Rusinkiewicz & Levoy 2001] provide good surveys over these ICP variants. Although these measures improve the convergence properties of the original ICP algorithms and achieve high registration accuracy, they still do not allow for a registration of several completely unaligned surface patches in reasonable time.

To automate the registration process, several authors proposed to detect special surface feature points on the surface patches [Faugeras & Hebert 1986; Kamgar-Parsi et al. 1991; Stein & Medioni 1992; Feldmar & Ayache 1996; Ashbrook & Fisher 1997; Johnson & Hebert 1997; Tarel & Boujemaa 1999; Sun & Abidi 2001; Higuchi et al. 2001; Krsek et al. 2002; Wyngaerd & Gool 2002; Yamany & Farag 2002; Sharp et al. 2002; Li & Guskov 2005; Gelfand et al. 2005]. Constraining the search for correspondences to these features can accelerate the registration process drastically and automatic registration becomes possible. Feature-based approaches primarily differ in their definition of feature points and in the way they are matched. A common drawback of these approaches is that they rely on a sufficient number of prominent or salient features in the geometry. Especially in the presence of noise or missing values this is often problematic.

To circumvent this problem Chen et al. [1999] developed a different approach: for pairwise registration they propose a randomized selection of control points on one of the surface patches followed by an exhaustive rigidly constrained search for corresponding points on the other surface. Robertson and Fisher [2002] also proposed an exhaustive search for automatic registration. Instead of searching for correspondences, they use a parallel search in pose space based on evolutionary algorithms. While the method of Chen et al. is sensitive to noise, the method of Robertson and Fisher requires relatively large overlaps in the surface patches in order to converge to the correct solution. Furthermore, both methods require substantial computational efforts.

Recently, Gelfand et al. [2005] and Li and Guskov [2005] proposed methods that derive and compare shape descriptors that either are inherently scale independent or explicitly scale space-based. That way, relevant features can be extracted robustly with respect to noise and the pre-alignment of range images can be performed considering the subset of distinguished points only. In this respect, these approaches are similar to the one presented here. However, one problem with feature extraction approaches that exploit the surface's geometry is that they tend to be unstable in the presence of missing (occluded) surface parts, which is particularly likely for single range images in the region of distinctive surface features.

Considering the desirable properties of image feature detection, it is not surprising that the idea of exploiting 2D-features for 3D-registration problems is not new. In [1999] Roth uses the popular Harris feature detector [Harris & Stephens 1988] to extract features from an intensity image that is aligned with a range image. Because of the large number of detected feature points, the author refrained from considering all possible feature point pairs for matching. Instead, the feature points of each surface in three space are tetrahedrised individually using a Delaunay tetrahedrisation and the search for correspondences is restricted to the faces of these tetrahedrisations. Two triangles are considered a match if their edge lengths match. However, due to occlusion and missing values in the range images, feature points might be present in only one of the two range images and the Delaunay tetrahedrisations become inconsistent. Therefore, the method is limited to relatively small view point changes and range images with only few missing values.

Another approach related to our method was presented by DePiero in [2003]. While his method is not based on image features, it detects KLT features [Lucas & Kanade 1981] in range images and maintains these features together with a graph structure in a database. Targeting at the fast registration of range image sequences, the method predicts the sensor movement from the previous images and uses this prediction to project a subgraph from the database into the next range image in the sequence. This predicted subgraph is then fitted against the detected features, and corresponding features are identified by a graph matching algorithm. While this approach is reported to register a range image sequence at rates of up to 10Hz on contemporary PC hardware, it relies on the viewpoint changes between subsequent images to be comparatively small. In addition, this approach also does not exploit the additional discriminative information contained not in the geometry but in the photographs typically recorded with the depth images.

4.1.2 Multiview Registration

If more than two range images are to be registered a simple solution is the incremental approach taken in [Besl & McKay 1992; Masuda et al. 1996] and [Sappa & García 2000]: From the set of unregistered patches U two patches are chosen and registered using a pairwise registration method. The two registered patches are then merged into a single patch which is put back into U. This process is repeated until the set U contains only a single surface patch. This incremental approach suffers naturally from the accumulation of local registration errors leading to possibly large global registration errors.

Therefore, several authors proposed to solve for the position and orientation of all patches simultaneously [Blais & Martin 1994; Bergevin et al. 1996; Schmitt & Benjemaa 1997; Eggert et al. 1998]. All of these approaches minimise the sum of squared distances between closest point pairs or the distance between a point and the tangent plane to the corresponding point as suggested in [Chen & Medioni 1992]. As correspondences are iteratively recomputed during the optimisation, these methods are computationally expensive. As a countermeasure, Pulli [1999] proposes using a generalisation of the so-called *concrete-mate* approach, where point-point correspondences remain fixed during the multiview alignment. Also, Cunnington and Stoddart [1999] discuss methods that solve the multiview registration problem in case of known point correspondences. In combination with a feature point detection and matching scheme, these approaches can also be used for automatic multiview registration. However, their sensitivity to noise especially in cases where only a small number of feature points can be found and matched, lead to the hybrid approach incorporating both feature point and closest point correspondences proposed in this chapter.

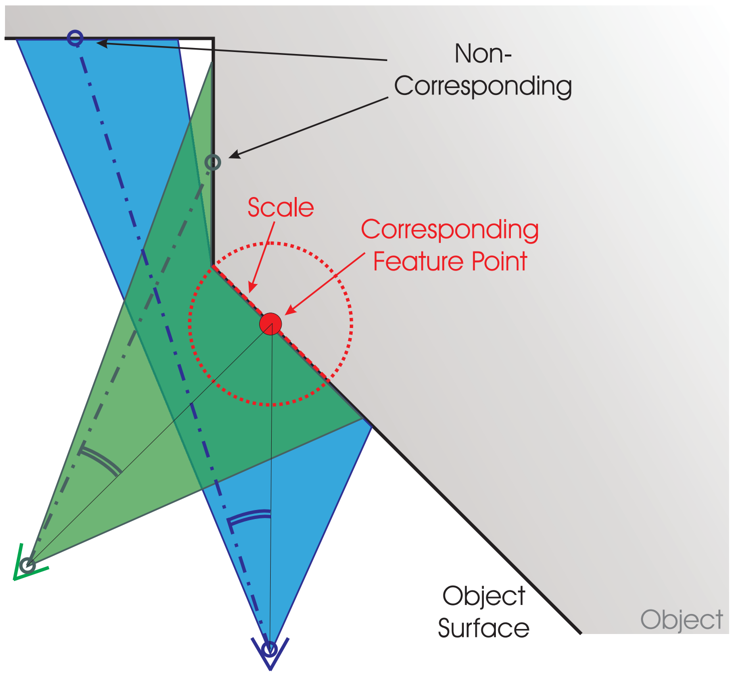


Figure 4.3: *Two range images (green and blue) with matching feature point and scale. Only inside the scale-induced* feature surface element *(red circle) the two range images can robustly be expected to contain corresponding parts of the object.*

4.2 Feature Detection and Matching

Finding geometric features in range images is non-trivial for several reasons. 3D-feature detection is already a difficult task for closed object representations and the situation worsens in case of surface patches acquired by digitising real-life objects. In this case, only parts of the object's surface are visible due to occlusion and limited field-of-view. Moreover, the fact that 3D-descriptors are naturally incapable of distinguishing local regions on surfaces of constant curvature (e.g. on planes, cylinders and spheres) makes this approach infeasible for many objects, in particular if they are geometrically highly self-similar or rotationally symmetric.

On the other hand, finding and matching features in 2D-images is a well-researched topic, and algorithms robustly detecting features that are insensitive even to brightness changes, scaling or local occlusions exist.

In a recent survey, Mikolajczyk and Schmid [2003] compared the performance of several local feature descriptors. In particular they examined the robustness of the features with respect to noise, lighting and view point changes up to 60 degrees. They found the *Scale Invariant Feature Transform* (SIFT) which was developed by Lowe [1999] based on earlier work by Lindeberg [1993] to perform best (see also [Lowe 2004]). Being based on a scale space representation of the underlying image, SIFT detects features with a scale parameter that reflects the spatial extension of its defining image neighbourhood. This scale property is of vital importance here since it allows to robustly estimate a 3D-position for each detected image feature.

4.2.1 Feature Surface Elements

Let I be the 2D photographic image acquired together with the depth image. Let

$$\chi : I \rightarrow \mathbb{R}^3 \cup \{\text{void}\}$$

be the known one-to-one correspondence between the pixels in the intensity image I and the depth values in the range image usually established during the data acquisition process.[1] Suppose further that $f \in I$ is a feature point in the 2D-image as detected using the SIFT features. Finally, let $I_f \subset I$ be the part of I supporting f. A straightforward approach to define 3D-features $\tilde{\mathbf{f}}$ would be to simply evaluate $\chi(f)$. Unfortunately, this straightforward approach is not commendable, since the resulting 3D-point $\tilde{\mathbf{f}}$ would be sensitive to noise and small feature deviations. Furthermore, and more severely, f might correspond to a place on the 3D-object where no geometry data has

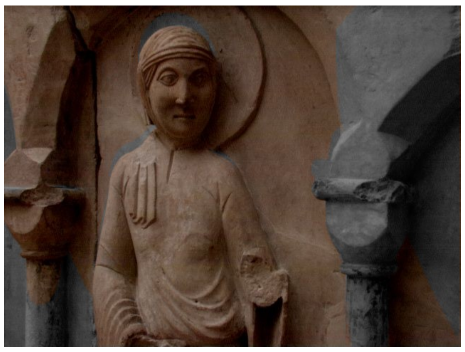


Figure 4.4: *Colour image with pixels greyed out that correspond to missing depth values.*

[1]As illustrated in figure 4.4, typically not all pixels in I do have a corresponding 3D position in the range image.

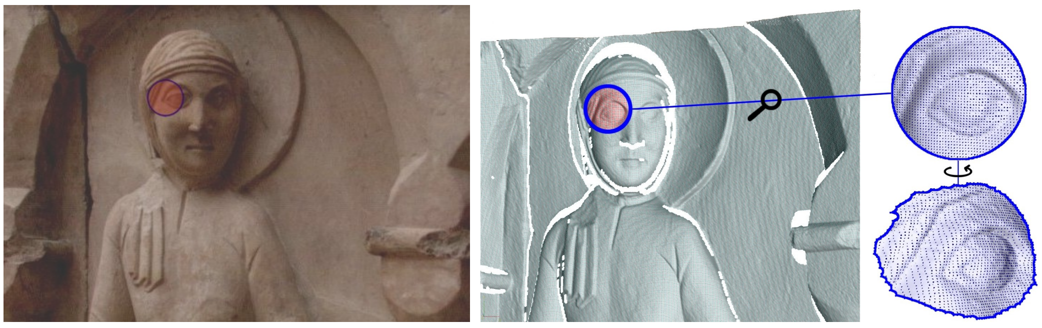


Figure 4.5: *3D-feature surface elements are derived from scale-equipped 2D-features.*

been acquired, e.g. in shadowed regions or at dark or reflective spots on the object's surface. This fact is illustrated in figure 4.4. The greyed-out regions in this image have corresponding 3D data, i.e. these pixels are χ-mapped to *void*.

Therefore, instead of using a single 3D-point (the direct corresponding point to the 2D-feature point) as feature, the set $\chi(I_f)$ of all points corresponding to the image area I_f determined by the position and scale of the feature is considered (see figure 4.5). These sets are called *feature surface elements* to accent that they are indeed a surface realisation of the scale-equipped feature points. Please note that the similarity to the notion of *surfels*, i.e. surface points equipped with normals, is not accidently: Surfels implicitly store a local first-order approximation of the neighbouring surface. Analogously, feature surface elements represent a sampling of the neighbourhood. Unlike surfels though, the feature surface elements represent a region on the surface with a well-defined size known from the 2D-image features.

According to the above definition, a *feature point* $\mathbf{f}$ is defined as the centre of gravity of the respective feature surface element

$$\mathbf{f} = \sum_{i \in I_f} \chi(i).$$

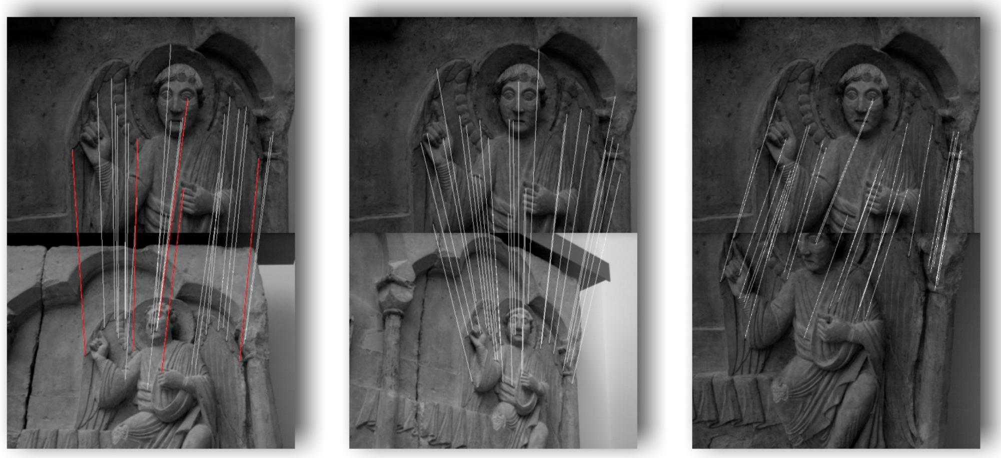


Figure 4.6: *Feature matches detected in image pairs (some matches highlighted in red for better visibility). The employed SIFT features [Lowe 1999] performed well despite considerable change in* perspective, scaling, intensity, *(left and centre) and despite only* small overlap *(right).*

4.2.2 Consistent Feature Matching

For any pair (ι, κ) of range images, let $\mathcal{C}_{\iota\kappa}$ denote the set of corresponding feature points. (see figure 4.6).

Although the SIFT method already provides good matching results, false positive matches are nevertheless possible. Since the subsequent registration steps are sensitive to such false correspondences, additional filtering is required for the matches based on the RANSAC method [Fischler & Bolles 1981].

A set of matching features in a pair of images can be validated as soon as the 3D-positions of the features have been determined. The basic idea is that overlapping regions of a pair of range images (ι, κ) represent the same part of an object. The *relative* positions of paired matchings must therefore be consistent. Thus, the validation can be reduced to checking their conformity with respect to rigid transformations, as illustrated in figure 4.7. Since it is computationally expensive to actually compute the largest conformal set of matching features (maximum clique), the RANSAC method randomly selects a set of three feature pairs and computes its support, i.e. the set of all feature pairs conforming to the

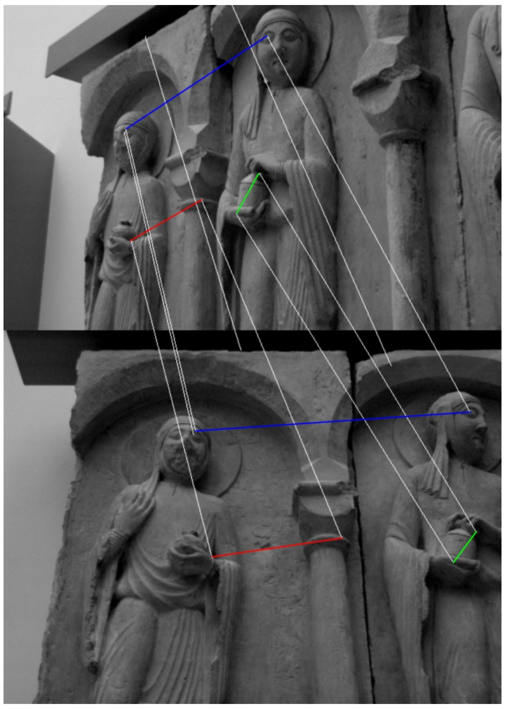


Figure 4.7: *Matching feature pairs in 2D are validated with the relative positions of their 3D counterparts (indicated by the red, green, and blue lines), which should be coinciding in the case of a* conforming *match.*

implied transformation. A support set is rejected if it is below a certain size (a value of 6 performed well for the results presented in this chapter). This way, unreliable correspondences are effectively removed since large sets of false, yet *conforming* matches are extremely improbable.

Although the 3D-feature point positions are stable with respect to noise, the sampling of a feature surface element in different images is usually not consistent. In addition to missing range values due to reflective spots, shadowing etc., other factors like a varying sampling density might lead to slight deviations in their 3D-positions. While such deviated features can be filtered out using the RANSAC approach to improve the registration accuracy, such deviations can also be tolerated to a certain extent to increase the number of conformal matches. This constitutes a trade-off between the connectivity in the registration graph (see section 4.4) and the accuracy. An additional constrained domain alignment

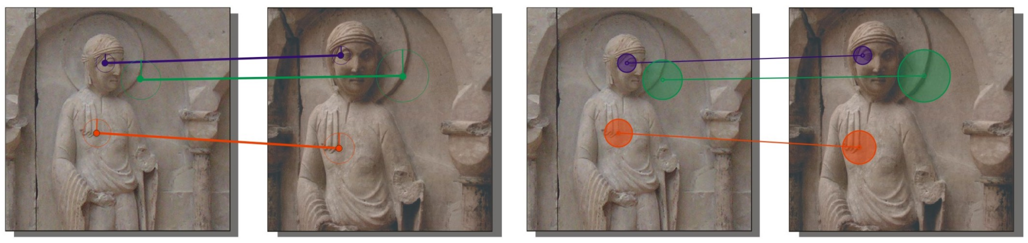


Figure 4.8: *In the first registration stage, only the centres of the feature surface elements are aligned (left). The next stage aligns all available corresponding points pairs contained in the feature surface elements (right). For illustration purposes, the conforming feature matches were highlighted manually in this figure.*

step described in section 4.3 compensates for the tolerated feature deviation.

4.3 Pairwise Registration

From the algorithm described in the previous section, we have for any range image ι a set F^ι of scale-equipped feature points $\mathbf{f}_i^\iota$, $i = 1, \ldots, n^\iota$. Moreover, for any pair (ι, κ) of range images we have a (possibly empty) set of correspondences

$$\mathcal{C}_{\iota\kappa} = \{(i,j) \mid \mathbf{f}_i^\iota \in F^\iota \text{ and } \mathbf{f}_j^\kappa \in F^\kappa \text{ corresponding}\}.$$

This section describes a two-stage registration procedure for a pair (ι, κ) with non-empty correspondence set $\mathcal{C}_{\iota\kappa}$ (see figure 4.8).

Coarse Registration

The first registration step consists simply of aligning the point sets F^ι and F^κ in a least squares sense, i.e. of finding (among the set of all rigid transformations) the solution to the local minimisation problem

$$T_{\iota\kappa} = \arg\min_T \ \varepsilon(T \cdot \iota, \kappa), \tag{4.1}$$

where the registration error ε is defined as

$$\varepsilon(\iota, \kappa) = \sum_{(i,j)\in\mathcal{C}_{\iota\kappa}} d^2(\mathbf{f}_i^\iota, \mathbf{f}_j^\kappa). \tag{4.2}$$

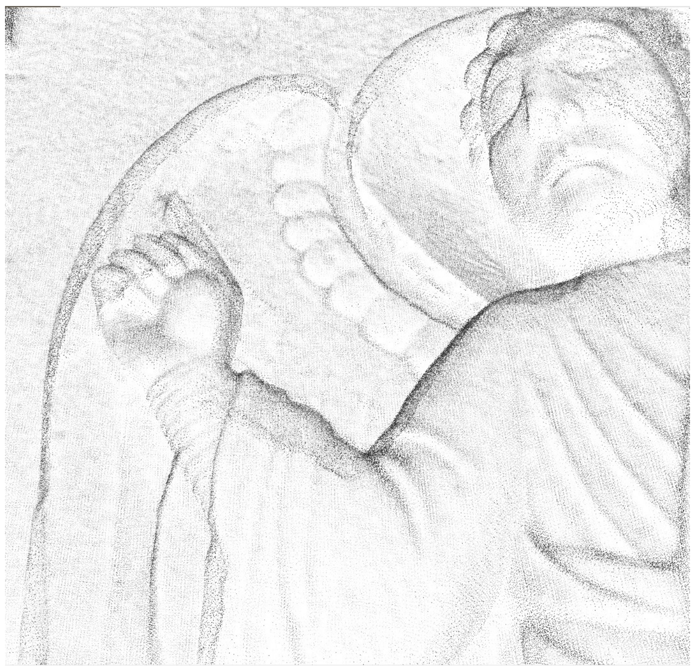


Figure 4.9: *Detail View of the reconstructed angel using 17 range images. Registration was performed on the feature points and the feature surface elements only.*

Since correspondences are known and fixed, this is a non-iterative procedure (in our implementation solved using the method described by Horn in [1987]), leading efficiently to an initial registration for ι and κ.

The alignment based solely on the feature points, however, accounts only for a fraction of the information available in the range images. (Typically, the number of conforming feature points is in the order of dozens compared to the several hundred thousands of data points.) To compensate for the errors induced in the feature point computation as described in the previous section, a second registration step is performed.

Fine Registration

Basically, it would be possible now to register the pre-aligned pair of range images applying one of the many variants of the ICP-algorithm. They have proven to lead to excellent registra-

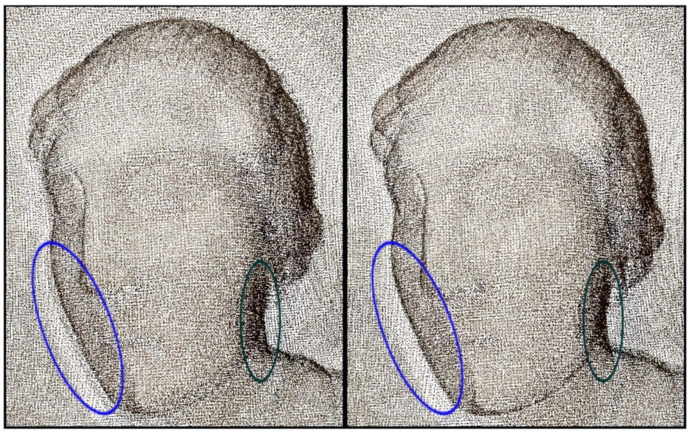


Figure 4.10: *Detail of the registered rood-screen before and after relaxation. In the left picture, registration errors are noticeable in the area of the chin, the cheek and in the neck.*

tion results for good starting positions. Unfortunately, they are computationally non-trivial and imperilled of false correspondence computation, which might lead to slow convergence and, more importantly, is susceptible to run into local minima as described in section 2.3. These problems (which are fundamental in nature) can be reduced considerably by restricting the domain for the correspondence computation to regions of the object that are *known* to correspond: From the feature detection in the 2D-images, we know that the feature surface elements introduced in section 4.2 constitute corresponding parts of the surface.

To align the feature surface elements, an ICP on constrained domains is performed: During the correspondence estimation phase of the ICP-algorithm, new correspondences are only obtained for the points representing the respective feature surface elements. For a pair $(i, j) \in \mathcal{C}_{\iota\kappa}$, these are the point sets $\chi(I_{f_i})$ (as *model*) and $\chi(I_{f_j})$ (as *data*) respectively.

Effectively, the range images ι and κ are thus aligned using well-known ICP-techniques but with the exception that only the scale-equipped feature surface elements are considered, thereby drastically reducing the risk of false correspondence estimation. Figure 4.9 shows a detail of the reconstructed rood-screen after the two-stage registration process.

In the above formulation, the 2D-feature matching procedure does not take into account the distribution of the feature points over the range images. In cases where the bounding box of the feature surface elements is very small compared to the bounding box of the range image itself, the two registration steps presented above might leave a registration error noticeable in regions far from the feature surface elements. In these cases, as a consequence of the high-quality pre-registration, a final ICP stage performed on the full data will resolve the remaining inconsistency. In many cases, though, (and in all the pictures presented in this chapter), the fine registration by feature surface element alignment is sufficient.

4.4 Multiview Registration

For non-synthetic data, the bilaterally optimal transformations will typically be non-conforming, i.e. the optimal transformation of a range image with respect to one other range image will not be optimal with respect to the remaining range images. To mediate between the competing transformations, a graph relaxation algorithm will be introduced in this section to solve the multiview registration problem.

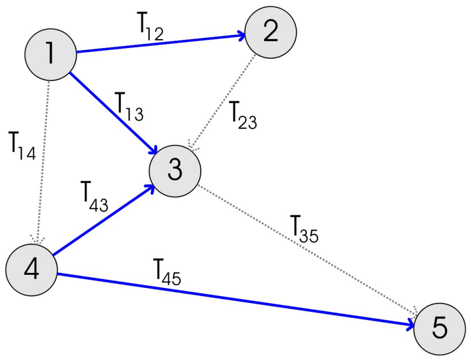


Figure 4.11: *The registration graph and a corresponding spanning tree. Setting T_3 to be the identity would give, e.g., $T_1 = T(1,3)$, and $T_5 = T(4,5) \circ T(4,3)$ as initial transformations.*

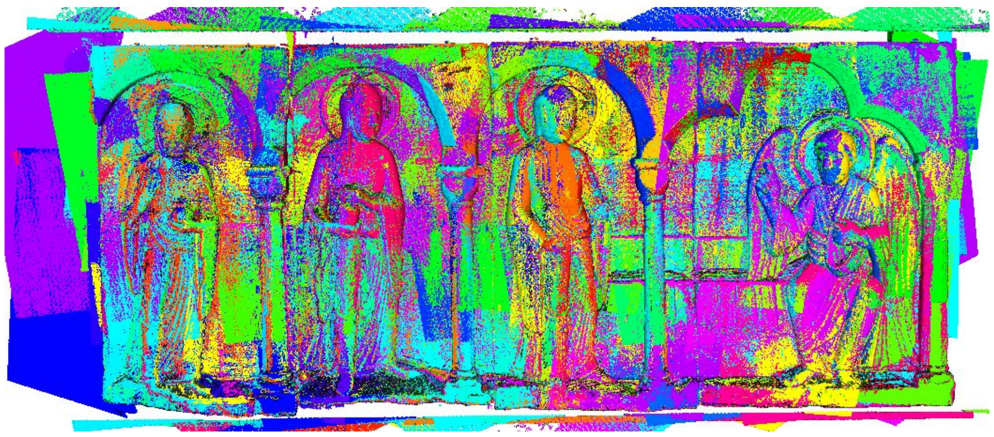


Figure 4.12: *Registered range images. 84 Patches, 20 million points. Note that many patches cover exclusively the interior of the object, a fact that would make the exploitation of synthetic marker points attached in the surrounding infeasible.*

4.4.1 Graph Setup

Let $\mathcal{G}$ be a directed graph $(\mathcal{N}, \mathcal{E})$. The nodes $\mathcal{N}$ represent the set of range images. An edge $e = (\iota, \kappa)$ is element of $\mathcal{E}$ iff the correspondence set $\mathcal{C}_{\iota\kappa}$ is non-empty. To every edge $e = (\iota, \kappa)$ we assign a rigid transformation $T(e) = T(\iota, \kappa)$ that is initialized to be the solution of the bilateral alignment process of the two adjacent range images. Additionally, we store with every edge the registration error $\varepsilon(e) = \varepsilon(\iota, \kappa)$ induced by this initial registration. The antisymmetry $T(\kappa, \iota) = T(\iota, \kappa)^{-1}$ in the edge attributes is the reason why $\mathcal{G}$ needs to be a *directed* graph – in all other respects $\mathcal{G}$ can be treated as undirected.

The task is now to find for every node ι a transformation T_ι such that the global registration error

$$\Sigma := \sum_{e \in \mathcal{E}} \varepsilon(e) \tag{4.3}$$

is minimal. In other words: Let $\mathbb{T}$ be the vector $(T_1, \ldots, T_n)$ of rigid transformations, then we're looking for the solution to the global minimisation problem

$$\boxed{\mathbb{T} = \operatorname{argmin} \sum_{(\iota,\kappa)\in\mathcal{E}} \varepsilon(T_\iota \iota, T_\kappa \kappa).} \tag{4.4}$$

4.4.2 Graph Collapse

Clearly, problem (4.4) has a degeneracy in the sense that the error Σ is invariant under any rigid Transformation Q:

$$\Sigma(\mathbb{T}) = \Sigma(Q\mathbb{T}) = \Sigma(QT_1, \ldots, QT_N)$$

Therefore, one can in principle choose an arbitrary node ι_0 s.t. T_{ι_0} is the identity transformation. An initialisation T_ι for all nodes ι can then be found by computing a minimal spanning tree of $\mathcal{G}$ and combining the transformations from ι_0 to ι along the paths in the spanning tree (cf. figure 4.11).

For numerical reasons it is beneficial to choose the root node ι_0 s.t. the average path length from ι_0 to all remaining nodes is minimal, otherwise the choice is arbitrary.

4.4.3 Relaxation

To resolve the non-conforming transformations at the graph nodes, the algorithm iterates over the original graph $\mathcal{G}$ and re-aligns each node with respect to the adjacent nodes. Again, this is a two-stage procedure: First, the relaxation is performed taking into account the feature points only, whereas in the second stage, the correspondences in the feature surface elements are accounted for.

In the literature, different approaches have been discussed concerning the recomputation of correspondences between iterations. Recomputing the correspondences between two iterations is not only computationally expensive, it might also exhibit slow convergence speed. This is due to the fact that changing the correspondences actually constantly changes the function to minimise. Moreover, since thresholding is applied during correspondence computation, the registration graph might even get disconnected in cases where subgraphs of the graph are connected only by very few cross-edges. Keeping the correspondences fixed during the whole relaxation, on the other hand, is sensitive to noise and prone to run into local minima. Hence, a hybrid approach is applied that keeps correspondences fixed during the relaxation and afterwards repeats the process with recomputed correspondences. In pseudo-code the relaxation reads:

```
relax($\mathcal{G}$,stage)
while $\Sigma$ improves do
    if stage $> 1$ then
        recompute correspondences;
    end if
    while $\Sigma$ improves do
        for all $\iota \in \mathcal{N}$ do
            align $\iota$ with adjacent nodes;
        end for
        evaluate $\Sigma$;
    end while
    evaluate $\Sigma$;
end while
```

Finding corresponding pairs as closest points results in asymmetric correspondence sets, i.e. $\mathcal{C}_{\iota\kappa} \neq \mathcal{C}_{\kappa\iota}$. This is appropriate in the *data-model* concept of registration, i.e. if one range image has to be aligned to another (since this relationship, too, is asymmetric). In multiview-registration, however, range images have to be aligned mutually. Otherwise, for an edge (ι, κ), a next relaxation step (where ι is the current node to be re-aligned) might simply try to undo the transformation just achieved in the last step (where κ was re-aligned), leading to slow convergence. Hence, the correspondence set for all edges $(\iota, \kappa) \in \mathcal{E}$ is defined to be the union of the *one-sided* correspondence sets $\mathcal{C}_{\iota\kappa}$ and $\mathcal{C}_{\kappa\iota}$. Obviously, this is not necessary in the first relaxation stage, where the correspondence sets consists only of the feature points themselves and, therefore, is symmetric by construction.

Furthermore, experiments show that during the pairwise registration it is typically sufficient to perform the first stage only, i.e. the alignment of the feature surface elements can be omitted in

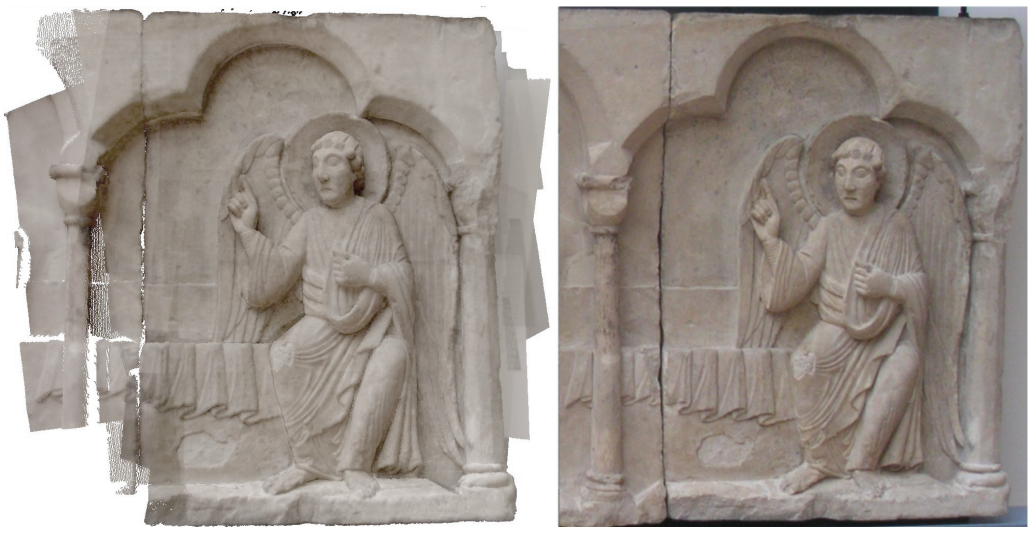


Figure 4.13: *Right: Detail photograph of the rood-screen. Left: Reconstruction of the detail; 17 range images were used, no global fine registration step applied. Note that the images were taken from a slightly different viewpoint. The colour difference mainly results from using a flashlight for the photography*

the bilateral case, and both stages need not be applied until the relaxation of the registration graph.

4.5 Results and Conclusions

Figure 4.12 shows the 84 patches that were registered to reconstruct the rood-screen depicted in figure 4.2 using the two registration steps described in sections 4.3 and 4.4. Figures 4.13 and 4.14 show point renderings of the reconstructed rood-screen. For the given examples, the complete registration process from feature detection and matching to the graph relaxation based on the feature surface elements took less than an hour on standard PC hardware and was performed without any user-interaction.

The key to the automatic registration of multiple images is the use of robust and expressive image features that additionally contain scale information. This extensive feature information allows us to perform a two-stage registration process in which a feature-

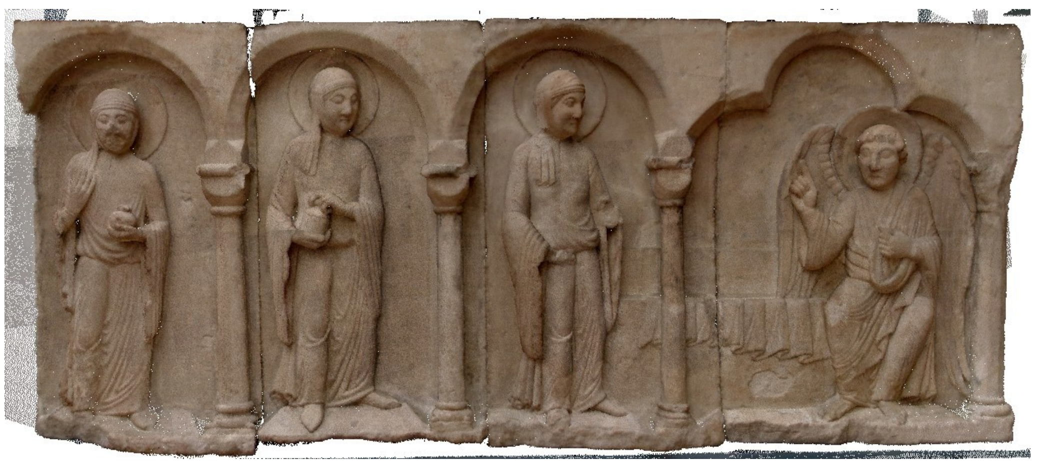


Figure 4.14: *Reconstruction of the complete rood-screen, point rendered with per vertex-colours*

point alignment precedes an alignment of feature surface elements. The latter is basically a constrained-domain ICP where the domains are consistently derived from the scales established in the 2D-feature detection and matching process. This approach scales well to large data sets and avoids local minima. The thresholds for the correspondence computation in the second registration stage are naturally derived from the registration error of the foregoing stage.

Despite its conceptual simplicity, the approach described above profits naturally from robust feature point correspondences. In particular, feature detection and matching on basis of 2D-images gives access to 3D-feature points at places infeasible using only the 3D-data, e.g. at concavities in the object, or spots on the object that do not deliver a 3D-point, but can easily and robustly be identified on the corresponding 2D-image. As a consequence, this approach is robust with respect to missing data in the range images due to the object geometry, material properties, or the scanning process itself, that were a major challenge in previous registration approaches.

Another important benefit of exploiting image-based features is that even surface patches that are geometrically indistinguishable can be robustly registered. Thus, rotationally symmetric objects can be reconstructed as well as objects that are highly self-similar if there is image information that can be evaluated.

In an approach very similar to the one presented here, Seo et al. [2005] applied perspective correction to the photographic images before the 2D feature matching phase, arguing that this improves the matching results. However, the considerable amount of feature point pairs typically detected for overlapping images, paired with the consistency check described in section 4.2, proved to deliver enough confident correspondences for the ensuing registration steps. Here, the algorithm also benefits from its hierarchical layout, where potential inaccuracies resulting from the feature point computation are remedied when the local patches, the feature surface elements, are aligned.

The above registration algorithm is independent of additional user-defined marker points – a point that is essential for cultural heritage applications, where artefacts often must not be touched at all. On the other hand, these marker points (if available) can naturally and easily be included in the registration process.

CHAPTER 5

Reconstructing Geometry from Dense Image Sets

There are two most notable reasons why laser range scanners are not under all circumstances the technology of choice for digitising 3D artefacts. Of them, one is more related to the source of the data, namely the artefact itself, and the other to the targeted later use of the data.
Concerning the artefact itself, it is obvious that not all materials of an archeological artefact (or any three-dimensional object in general, for that matter) are equally well-suited for digitisation using laser-range scanners. By construction, smooth, fairly diffuse materials are definitely on the pro-side for this type of digitisation, whereas shininess, transparency and high surface complexity constitute sometimes unsurmountable challenges, necessitating other types of digitisation.

On the other hand, the end product of a data acquisition project for many applications might be a simulated view only, possibly under synthetic lighting conditions. The emphasis in these applications is hence on *visual* rather than *metric* accuracy.

For this reason, this chapter discusses capturing and representing three-dimensional artefacts using dense sets of photographic images.

5.1 Motivation

Three-dimensional digitisation using laser range scanners has a long tradition in industrial applications, such as reverse engineering, quality control and part inspection, where the geometric error of the acquired data compared to the original model is the ultimate criterion. More recently, with hardware becoming available at more reasonable costs, the use of 3D data acquisition has expanded to other application fields like cultural heritage and entertainment industries. In these application fields, however, the faithful reconstruction of the visual appearance is often much more important than the geometric error alone.

In archeological and museum applications, where digital archives of cultural heritage artefacts become more and more commonplace, the demand for realistic reconstruction of an artwork's material properties *beyond shape* is particularly high, as this often gives critical hints about important characteristics of the object, such as for instance the location, epoch, or circumstances under which the artwork was created.

A standard approach to digitise a 3D cultural heritage artefact would be to acquire the geometry first (up to a certain degree of detail) using a Laser Range Scanner, structured light, tactile sensors or even a volumetric approach on the basis of MRT- or CT-data. Given an appropriate parametrisation, additionally recorded 2D-images can then be projected onto the geometry as diffuse textures to represent colour and fine detail information — at the disadvantage that the illumination is fixed to the conditions under which the photographs were taken. Fixing the lighting conditions, unfortunately, severely limits the use of 3D digital representations in cultural heritage applications, because moving, rotating and seeing an artefact in different environments is an important operation that often gives additional insight. A prominent and evident example, where a fixed light direction causes vital details to remain hidden from the user are cuneiform tablets captured by Malzbender et al. [2001]. Facing the problem that many of the tablets' intricate details are visible only under certain illuminations, they introduced so-called *Polynomial Texture Maps*, by which objects can be viewed under new incoming light directions. The view point, however, still has to remain fixed.

The fixed lighting and viewing conditions are also the main reason why traditional texturing methods are not capable of transporting the characteristic appearance of complex, inhomogeneous, yet often encountered materials like fabric, leather, wood, or metal.

Despite recent successes in automating the data acquisition process even with laser range scanners (e.g. the registration approach described in the previous chapter), geometry acquisition still involves sophisticated procedures during, and after, recording (such as path planning in order to ensure detection of the complete surface, reconstruction and parametrisation). As a consequence, 3D photography is still often perceived by end-users in the museums as a complicated and time-consuming procedure – apparently *impractical* for large classes of objects and applications.

Before this background, this chapter describes a novel high fidelity acquisition system that exploits dense image sets acquired with the multi-camera array developed by the computer graphics group at the University of Bonn (see figure 5.3) to synchronously capture an object's 3D geometry *and* material properties in a very time-efficient and user-friendly way. The images are analysed to reconstruct an artefact's coarse to medium scale geometry using a GPU-based visual hull technique, resulting in a closed triangle mesh. In parallel, the images are also used to capture the object's appearance into so-called bidirectional texture functions (BTF) – a 6-dimensional texture representation introduced by Dana et al. [Dana et al. 1997] which extends the common textures by dependence on light- and view-direction, and thereby allows for photorealistic rendering of an object's micro- and mesostructure (cf. Figure 5.1). As result, in this chapter a system is described that

- fully automatically acquires 3D-data, capturing an object's geometry *and* its visual appearance in form of bidirectional textures

- generates a faithful image based representation of the object's mesostructure using BTF-techniques and therefore effectively overcomes the limited accuracy of the visual hull technique

- is time efficient and very easy to use.

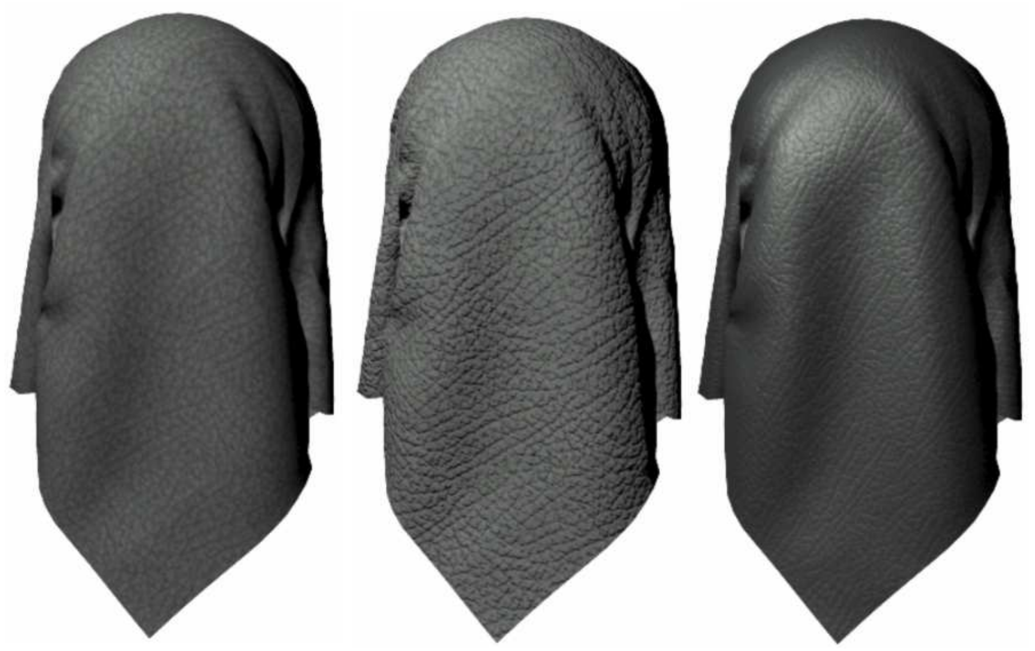


Figure 5.1: *Leather Material, rendered using standard texturing (left), bump mapping (middle), and BTF-rendering (right).*

The reasoning behind the decision to also reconstruct the explicit geometry (compared to a purely image-based and geometry-free approach) is that an efficient compression of the 6D reflectance field without geometry is impractical. In addition to this, direct access to the geometrical properties is a virtue in itself for various purposes, such as statistical shape analysis, interaction with other objects (*collisions, shadow casting*), multimodal interaction (*sound, haptics*), and modelling applications (*surface editing*).

5.2 Related Work

Defining models that allow for synthetic, photo-realistic images of objects to be rendered under new lighting conditions and for new view points is a well-established research topic in computer graphics. Traditionally the geometry of a surface is modelled explicitly (e.g. with triangles) only up to a certain scale, while the remaining surface properties responsible for the reflectance behaviour of

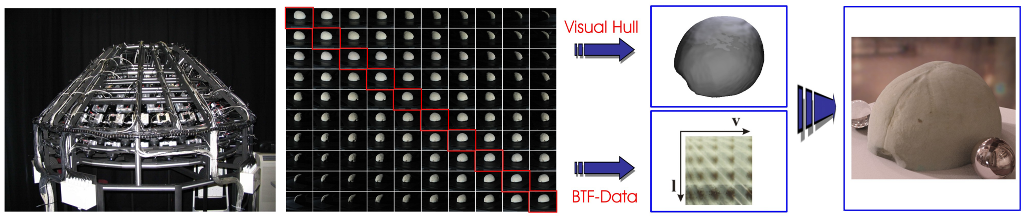


Figure 5.2: *The Acquisition Pipeline: The multi-camera array records 151 images per camera per light direction, resulting in 22801 images. Here, only the first ten cameras (from left to right) and the first ten light directions (from top to bottom) are shown. From the full set of these images, the BTF is constructed, while only a subset (the diagonal in this matrix notation) is used for the geometry reconstruction.*

a material are simulated using relatively simple analytical bidirectional reflectance distribution function (BRDF) models (e.g. [Lafortune et al. 1997]). Fine textural surface detail is typically captured by photographic images or procedurally generated textures which are projected onto the geometry.

Although this traditional approach has produced remarkable results for numerous examples, it becomes unfeasible for large classes of materials where the so-called mesostructure is essential for faithfully reproducing characteristic material properties.

Numerous researchers have proposed image-based methods to overcome these hindrances. Independently, Gortler et al. [1996] and Levoy and Hanrahan [1996] introduced light-field rendering, an efficiently renderable 4D representation of the plenoptic function. In their approaches, an object's complete appearance is stored in a four-dimensional function (called *Light Field* in [Levoy & Hanrahan 1996] and *Lumigraph* in [Gortler et al. 1996]), that describes the flow of light at the sampled positions in the sampled directions. These approaches allow photographic images – interpreted as a sampling of the complete light field – to be interpolated to generate images from new viewpoints; yet, the illumination has to remain constant.

To overcome this drawback, several researchers proposed representing an object in terms of its reflectance field [Debevec et al.

2000], [Hawkins et al. 2001]. Recently, Hawkins et al. also proposed a novel, dual way of measuring an object's reflectance properties [2005]. In particular [Hawkins et al. 2001] demonstrated remarkably realistic synthesised images of cultural heritage artefacts under new lighting conditions, although the generation of novel viewpoints remained basic.

In the above approaches, no explicit geometry is present, only renderings from new view points and/or under synthesised lighting conditions are feasible. In recent years, though, researchers have found the classical two-fold representation of an object as *geometry* + *material* to be superior in terms of time and memory efficiency [Wood et al. 2000][Coombe et al. 2005][Pulli 1997]. The approach presented in this chapter also relies on such a two-fold representation but our material representation captures both light *and* view dependent appearance variation. Therefore, the present approach is in concept most similar to the methods of Furukawa et al. [2002] and Lensch et al. [2003], who also construct a representation that allows rendering objects from novel viewpoints under arbitrary lighting. However, as they employ laser range scanners to record the 3D geometry of the object, still large classes of objects, in particular those with complex reflectance behaviour, can not be handled.

Moreover, the present approach uses an array of fixed cameras with fixed light sources, all mounted on a hemispherical gantry. Although similar acquisition setups have been used by numerous research groups, see e.g. [Matusik et al. 2002], [Furukawa et al. 2002], [Hawkins et al. 2001], the setup developed and built at the University of Bonn benefits from massive parallelisation and the fact that no moving parts (aside from the internal camera optics) are required – a fact that renders time consuming tasks like recalibration, registration, etc. unnecessary.

The simultaneous reconstruction of geometry and appearance is also an inherent part of the numerous methods aiming at 3D-reconstruction from uncalibrated image-sequences (e.g. [Pollefeys et al. 2004]). But these techniques are neither designed nor capable of performing the highly accurate reflectance measurements that are possible with a fully computer controlled and calibrated measurement device.

5.3 Overview

The basic concept of the present algorithm is as follows: In the acquisition phase, a dense set of photographs is recorded from predefined viewpoints, which are regularly sampled over the hemisphere. For each viewpoint $v_1, \dots, v_m$, we illuminate the object from light sources likewise regularly positioned at $l_1, \dots, l_n$. This recording results in a set $\{I_{ij}\}_{i=1 \dots m, j=1 \dots n}$ of $m \times n$ Images, where I_{ij} denotes the image taken from viewpoint v_i of the object to be digitised under illumination from the light source at position l_j.

With these images at hand, the first step is to reconstruct a triangulated surface representation of the object. To this end, a novel GPU-based variant of the classical volume carving method is applied to transform the visual hull derived from the dense set of images to a volumetric representation, from which the final triangle mesh is extracted. For this step, only the subset $\{I_{ii}\}_{i=1 \dots m}$ of the total set of images is exploited, i.e. only those images where light and viewing directions are coincident.

After parametrisation of the resulting triangle mesh, the next step is to exploit the full set of images to define the BTF. Efficient extraction, compression and storage of the BTF data is described in section 5.6. Figure 5.2 illustrates the whole process.

5.4 Multi-Camera Grid

The dense image set required to measure the BTF and to reconstruct the geometry of the object to be digitised is recorded using the multi-camera array described in [Koch 2006], which features 151 commodity digital still cameras mounted on a hemispherical gantry (see figure 5.3). Although similar gantries with mounted light sources have been used before (e.g. by Malzbender et al. to capture *Polynomial Texture Maps* [2001]) and other, more sequential setups (such as described in [McAllister 2002] or [Sattler

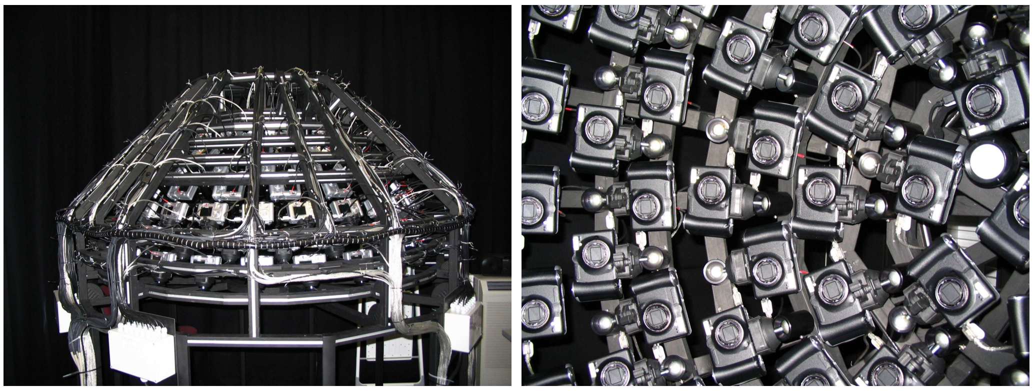


Figure 5.3: *151 commodity class, off-the-shelf cameras mounted on a hemispherical gantry. Right: Detail view of the dome. The camera fronts have been coated with diffuse black synthetics to reduce scattered light effects.*

et al. 2003]) could also be used, the setup at the University of Bonn is particularly suited for this technique due to its fast and calibration-free performance.

By arranging the cameras into this array, the acquisition of the images is parallelised and no moving parts (e.g. a rotating stage, a moving light source or camera) are needed. Therefore, the positions of the image sensors and the light sources can be calibrated in a preprocessing step which only has to be carried out if a camera has been replaced or after the whole setup has been transported. The low-level post-processing (geometric correction, colour correction) is fast enough to be done in parallel to the measurement.

For non-planar objects, the hemispherical setup can only deliver a subsampling of the full set of possible light and viewing directions. Figure 5.4 illustrates this effect. For geometry and appearance acquisition, however, this subsampling can easily be completed to cover the full sphere by repositioning the object. The registration of the recorded images after repositioning with the original images can be performed with the automatic registration technique given in the previous chapter (chapter 4). It is worth noting that with a fully automatic technique at hand, this repositioning approach can also be used to deliberately increase the sampling rate, which would otherwise be limited by the finite

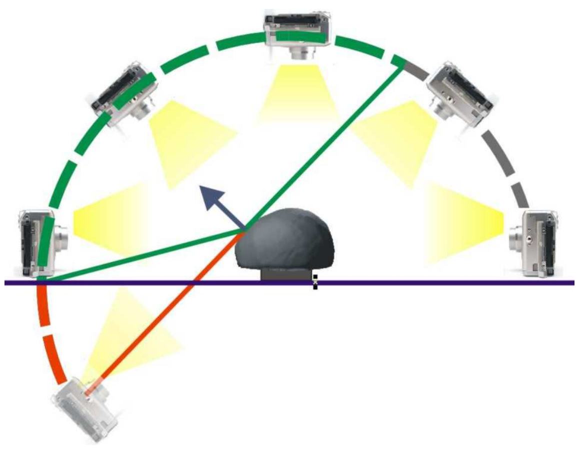


Figure 5.4: *Hemispherical setups – in contrast to full spheres of cameras and light sources – produce an incomplete subsampling of the full set of possible light and view directions.*

number of cameras mounted on the gantry. In its current setup, the gantry of the computer graphics group at the University of Bonn is equipped with consumer-class, off-the-shelf cameras with an image resolution of 2048 × 1536 pixels, coated with a diffuse black finish to reduce scattered light effects in the images and controlled by a network of commodity personal computers. As light sources, the built-in flash lights of the cameras are used, no further light sources are required. Formally, we therefore have $n = m$ and roughly $v_i = l_i$ for all $i = 1, \ldots, n$ in the above formulation. As consequence of the massively parallelised setup, a full dataset of $151 \times 151 = 22801$ images is acquired in only about 40 minutes.

5.5 Geometry Acquisition

One of the major advantages of our acquisition device without any moving parts is that camera and light source positions are known a-priori, as are hence the transformations required to transfer the different views into a common, global coordinate system. Thus, the registration of the multiple views from one set of images is straightforward.

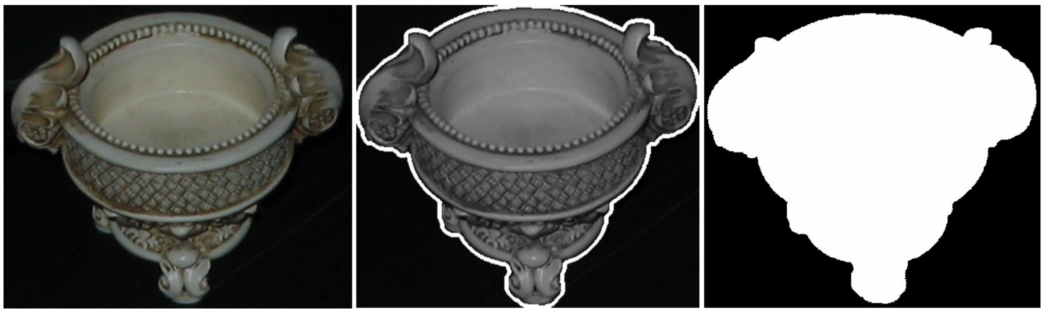


Figure 5.5: *One example image* $\in \{I_{ii}\}_{i=1,\dots,n}$ *(with coincident light and view direction), the extracted silhouette, and the corresponding binary* inside/outside-*image after thresholding.*

5.5.1 Visual Hulls

The idea to exploit a set of registered photographs of an object to derive its three-dimensional layout is far from new [Baumgart 1974]. Due to its conceptual simplicity and availability – only classical 2D photographic images are required – numerous authors have developed algorithms to reconstruct the 3D geometry of an object from images, long before 3D acquisition devices as laser range scanners have become commonly available. Here, a volumetric approach in the spirit of *shape from silhouette* [Martin & Aggarwal 1983] is applied. In many cases, the object's silhouettes can be extracted from the 2D-photographs via simple thresholding. By construction, a backdrop is not available in the hemispherical setup, since a backdrop would occlude at least part of the possible light directions for any view direction. Instead, the black coating of each component of the setup combined with the directed flash light sources focussed on the object to be digitised allows for the following approach: Every pixel with a brightness of less then a certain threshold is said to be *outside*, the remaining pixel are *inside* (see Figure 5.5). Obviously, this approach is only valid if most of the object's surface visible to the current camera is well-lit. Consequently, only those images are exploited where light and view directions are identical, i.e. $\{I_{ii}\}_{i=1,\dots,n}$, resulting in a set of binary images $\{J_i\}_{i=1,\dots,n}$. For objects, for which this thresholding approach is not quite sufficient, in particular for darker and more specular appearing objects, graph-cut image segmentation is applied with only very basic user-interaction, and finally difference images. In this last step, we again exploit the fact that all cameras are fixed in a static array, such that images can be taken once with and once without the object in place.

Combined with the viewpoint position (known a-priori from the setup), every outside-pixel in each image defines a ray in scene space that is known *not* to intersect the object, whereas the inside-pixels define rays that intersect the surface at some unknown distance to the viewpoint. In the continuous case (*pixel width* $\rightarrow 0$) the union of all these intersecting rays would define a generalised cone that is guaranteed to contain the object.
As this fact holds for all acquired images, the intersection of all generalised cones (named the *Visual Hull*, [Laurentini 1994][Matusik et al. 2000]) describes a tight volume in space in which the complete object must lie.

We make use of this guarantee by applying volume carving to a regular grid, i.e. we (conceptually) traverse the grid following the *outside*-rays and mark any cell that is encountered during traversal as *empty*. The triangle mesh is then constructed using the well-known marching cubes approach (see section 2.4.4).

5.5.2 Efficient Evaluation on the GPU

The large number of acquired images and the (potential) need for finer grids make it impractical to actually traverse the grid following the *outside*-rays. Instead, a hardware-supported approach based on projective texture mapping can be used:

Suppose, we have an axis-parallel grid of dimension $X \times Y \times Z$, corresponding to the axis directions $\mathbf{e}_x$, $\mathbf{e}_y$, and $\mathbf{e}_z$, respectively. For reconstruction, each grid point has to be assigned either the value of 0 (for outside) or 1 (for inside). Let $\mathbf{v}_i$ be the viewing direction from viewpoint v_i onto the object, and let (without loss of generality) $\mathbf{e}_x$ be the direction such that the scalar product $|\langle \mathbf{e}, \mathbf{v}_i \rangle|$ is maximal for $\mathbf{e} \in \{\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z\}$.

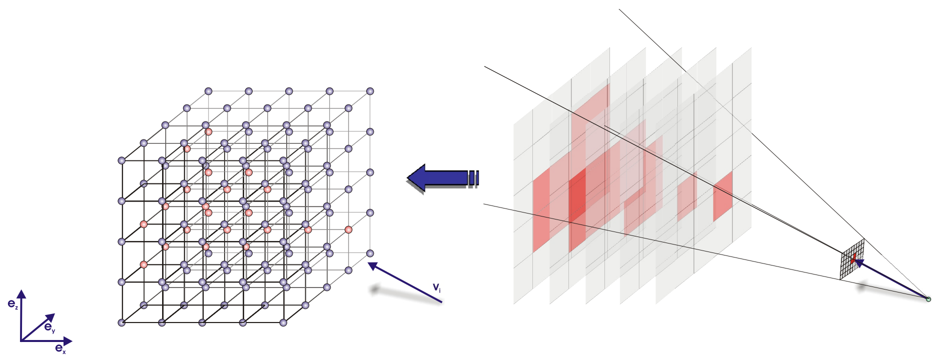


Figure 5.6: *The 3D grid is interpreted as set of binary images.*

We interpret the grid to be a stack of binary 2D textures $\{T_k\}_{k=1,\ldots,X}$, where each T_k has a resolution of $Y \times Z$ pixels, see Figure 5.6. The *inside/outside*-information is then efficiently collected by projecting every source image J_i to each texture T_k. We perform bitwise `AND`-operations during this projective texture mapping, to set a pixel in F_k to 0 if at least one J_i indicates so.

5.6 Bidirectional Texture Functions (BTF)

The image-based approach to capturing the appearance of an object for later rendering is to take dense sets of images under controlled viewing and lighting conditions in order to sample its reflectance field appropriately. As mentioned, Malzbender et al. [2001] captured the lighting variability of a texture by taking images lit from different directions and compressed the data to a compact representation called Polynomial Texture Maps. In order to be also able to vary the viewing direction, *BTFs* can be employed. Mathematically, the BTF can be expressed as a measured 6D-slice of the general 8D-reflectance field

$$\mathbf{RF}_{rgb}(\mathbf{x}_i \to \mathbf{x}_o, \omega_i \to \omega_o)$$

parameterised over a base surface S:

$$\mathbf{BTF}_{rgb}(\mathbf{x}, \omega_i \to \omega_o) = \int_S \mathbf{RF}_{rgb}(\mathbf{x}_i \to \mathbf{x}, \omega_i \to \omega_o) d\mathbf{x}_i$$

Please note that fixing the position **x** restricts the BTF at this position to a 4-dimensional BRDF, which is often called *apparent BRDF*, because this function still contains local self-occlusions, scattering, etc. [Müller et al. 2005].

5.6.1 Compression

One major issue of high fidelity BTF measurements are the huge memory requirements. A raw BTF data set easily requires multiple Gigabytes of memory, requirements that cannot be fulfilled even by the most state-of-the-art graphics hardware. Therefore, BTF data is usually compressed either by fitting standard BRDF models to the data or by using statistical analysis methods, such as principal component analysis (PCA) or clustered PCA. The latter is used here, and the resulting data can easily be decompressed and evaluated using graphics hardware [Müller et al. 2004].

Statistical analysis, however, requires that data entries in the BTF are semantically correspondent, i.e. that corresponding data entries in the BTF belong to the same physical point on the object. Unfortunately, this is a prerequisite that is fulfilled for the raw data only under the assumptions of planarity, orthographic projection and directional light sources. This is not the case here — among other reasons, since the dimensions of our acquisition setup cannot be considered "large" compared to the dimensions of the objects to be digitised. Therefore, the raw BTF data is resampled before compression based on a planar *parameterisation* of the reconstructed triangle mesh.

5.6.2 Parameterisation

In order to enable compression of the BTF data using statistical analysis methods, but also in order to enable simple textured rendering, the reconstructed geometry has to be parameterised. Surface parameterisation is an established field of research and although our geometry is generated using a visual hull algorithm and therefore some special properties of the surface could be exploited for a special parameterisation algorithm, we refer to the extensive

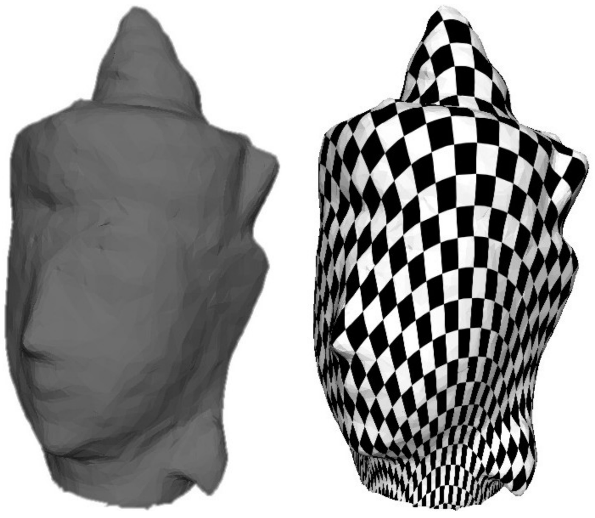


Figure 5.7: *Angkor Vat bust. Acquired geometry with simple Lambert shading and checker board textured to illustrate the parametrisation.*

work that has already been done in this field. Most importantly, our algorithm is not dependent on any specific parameterisation algorithm, and although demands like conformity and efficient packing do apply here, any available parameterisation algorithm could be used. For a good survey, see [Floater & Hormann 2005]. In our case, we use the parameterisation described in [Degener et al. 2003] (cp. figure 5.7).

5.6.3 Incomplete BTF data for non-planar objects

To capture the reflectance of a material independently of the specific geometric layout, most common approaches for BTF acquisition record images of a small planar material sample. Then a projection of the images on a common plane typically suffices. For digitising 3D artefacts, though, planarity cannot be presupposed. Hence, effects like self-shadowing and occlusion have to be dealt with.

Measuring BTFs generally consists of recording for every point on the surface its reflectance from each view direction under each light direction. For non-flat surfaces, however, the reflectance for some light and viewing directions will be zero (or close to zero)

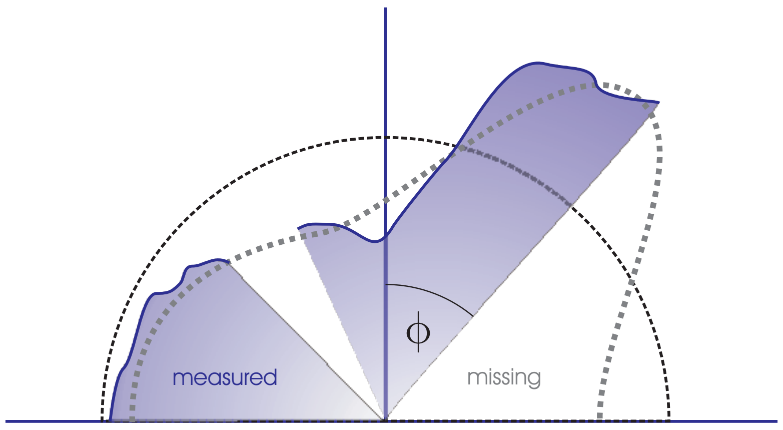


Figure 5.8: *Polar plot illustration of the measured BTF, that might be incomplete for some directions Φ due to occlusion and has to be completed, e.g. using statistical analysis methods.*

simply because of occlusion and/or self-shadowing. Using standard approaches, this *missing* information would be misinterpreted as a property of the material, a fact that necessitates a different technique to interpolate the occluded data.

First, from all the points on the object surface those points are identified that have an incomplete BTF measurement, i.e. points which are occluded for at least one light source or camera position. Figure 5.8 illustrates this situation.

For these points, the BTF has to be completed. One approach, that turned out to be very successful for the materials investigated so far, is to perform statistical analysis, here LPCA, of the measured BTF of all object surface points for which this is complete. The eigenvectors to the largest eigenvalues of the corresponding covariance matrices span a lower dimensional subspace approximating the original measured data. This way each individual bidirectional reflection function $\mathbf{BTF}_{rgb}(\mathbf{x}, \omega_i \to \omega_o)$ in a surface point $\mathbf{x}$ can then be approximated by

$$\mathbf{BTF}_{rgb}(\mathbf{x}, \omega_i \to \omega_o) = \sum c_k u_k,$$

where the u_k are the basis (apparent) BRDFs and the c_k the corresponding weights.

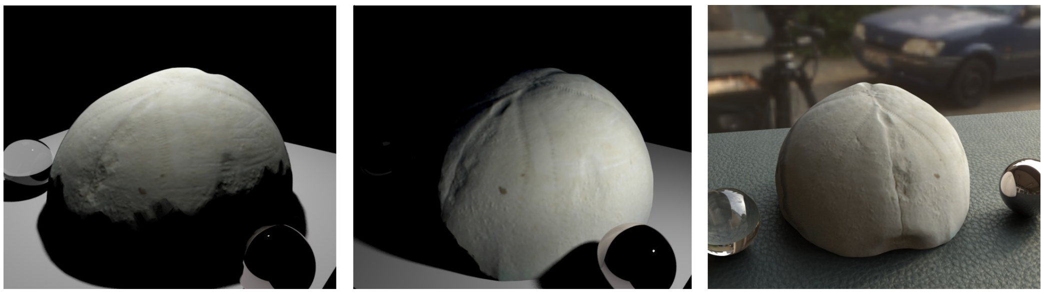


Figure 5.9: *Raytraced renderings of a captured and reconstructed echinite under novel lighting and viewing conditions. The left and middle image are rendered with a small area light source and demonstrate the fine geometric details captured in the BTF. The right image shows a relighting of the echinite with a complex image-based lighting environment captured in front of the Pädagogische Fakultät in Bonn. The ground floor in the right image is covered with a synthetic leather BTF courtesy of DaimlerChrysler AG.*

The incomplete BRDFs are completed individually, depending on the specific values that are missing. Let π be the projection of a full BRDF-vector $\mathbf{BTF}(\mathbf{x})$ to a lower dimensional subspace that only contains the *present* measurements and neglects the missing values. The next step is to find a set of coefficients $\{c_k\}_{k=1,\ldots,K}$ such that

$$\|\pi(\mathbf{BTF}(\mathbf{x})) - \sum c_k \pi(u_k)\|$$

is minimal. The reconstructed vector $\sum c_k u_k$ is a reasonable completion of $\mathbf{BTF}(\mathbf{x})$. For complex materials this process can be iterated, while taking into account the already completed vectors.

5.7 Results and Conclusions

The presented approach to reconstruct an object's geometry from the acquired images using visual hulls computed on the GPU is reliable and fast. Of course, identifying a non-convex object using a silhouette-based approach inherently and inevitably implies neglecting some features of its surface geometry. Despite this seemingly general inaptness of the visual hull reconstruction, realistic images of captured objects can nevertheless be synthesised be-

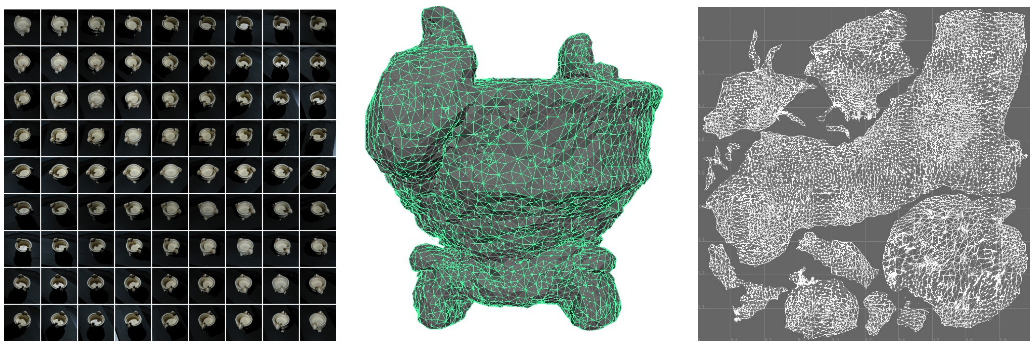


Figure 5.10: *From left to right: 9×9 subset of the recorded images of a vase, a (decimated) triangle mesh representation of the reconstructed geometry and the texture atlas, automatically computed according to [Degener et al. 2003]*

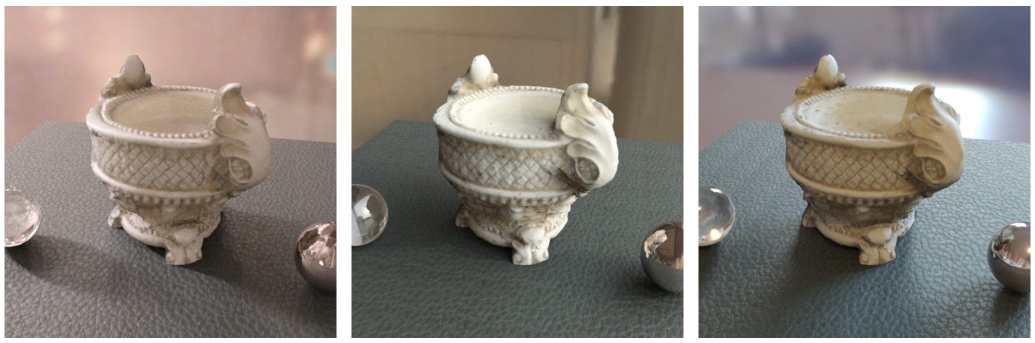


Figure 5.11: *Raytraced renderings of the above geometry with the captured BTF under various lighting conditions*

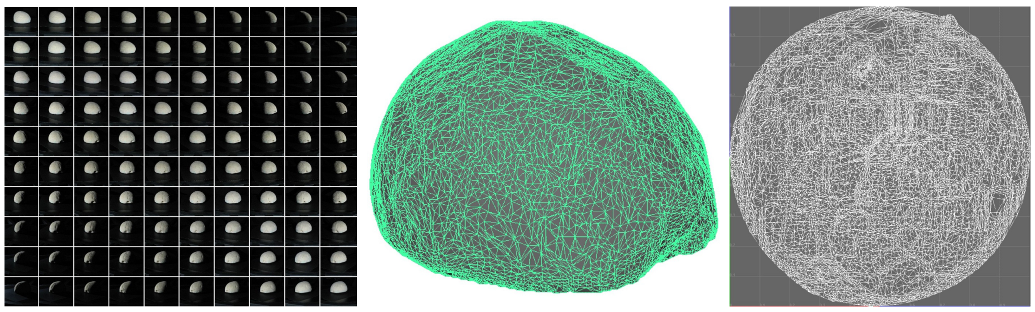


Figure 5.12: *From left to right: 10×10 of the recorded images of an echinite, the reconstructed geometry and the texture atlas used to create the images in figure 5.9.*

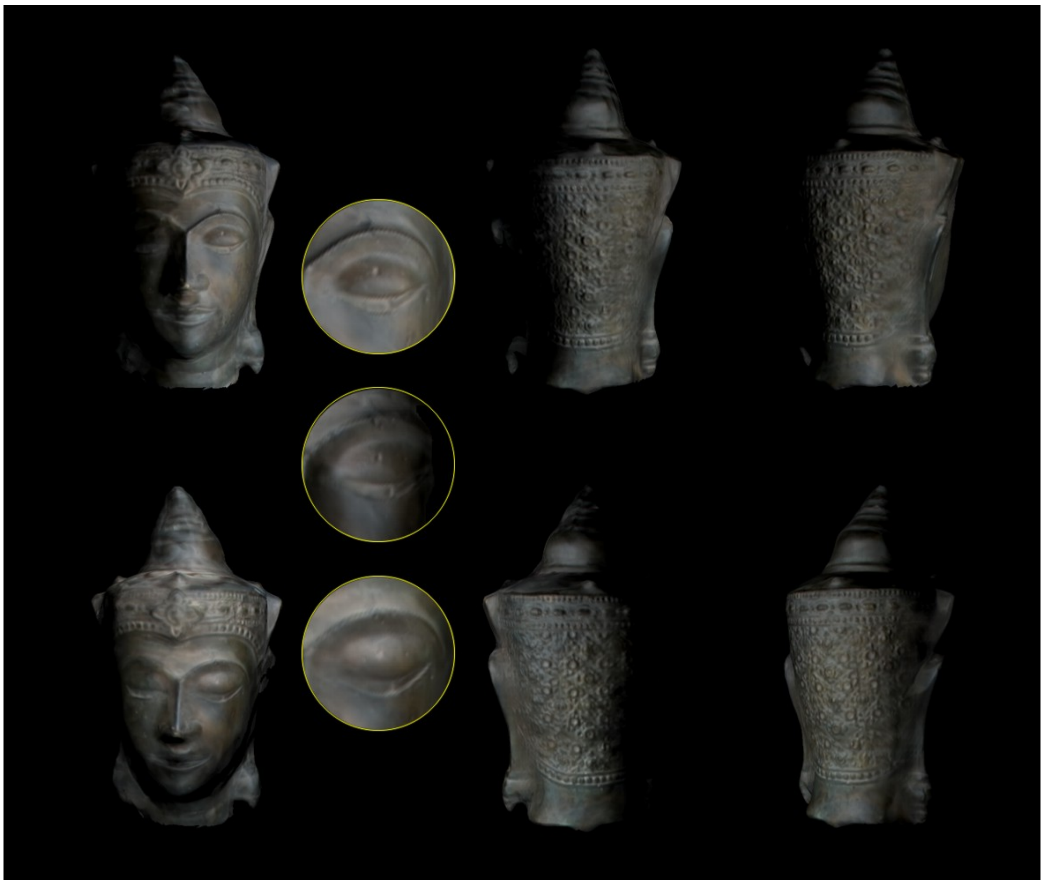


Figure 5.13: *Point light renderings of the Angkor Vat bust under varying light and view directions. This image illustrates the effect that fine geometry detail (like the defect in the bust's eye, see the zoom-ins) often remains hidden until seen from a specific direction with a specific light direction.*

cause the neglected surface features are well-captured in their appearance using the BTF texturing techniques. Figures 5.9 and 5.13 demonstrate this effect. The defect on the front side of the echinite in 5.9 as well as the defect in the left eye of the Angkor Vat Bust (see images 5.13 and 5.14) is not detected by the visual hull algorithm and therefore not represented in the geometric reconstruction. Their appearance is nevertheless preserved. The same effect can also be observed in figures 5.10 and 5.11; also here, on the basis of a very coarse geometric reconstruction, mesostructure details like the fabric pattern on the vase's rim are well represented

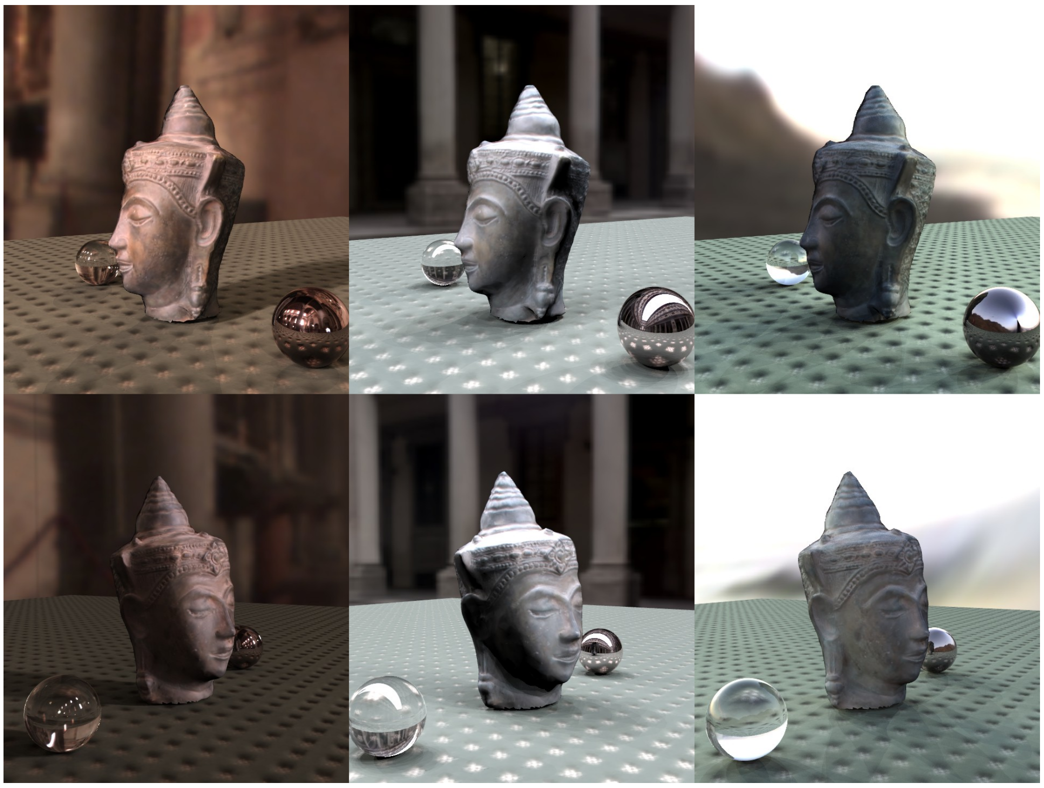


Figure 5.14: *Raytraced renderings of the Angkor Vat bust under various environmental high dynamic range illuminations. From left to right: Under the illumination of Galileo's tomb, Uffizi Gallery, and Beach Sunset. HDR images courtesy of Paul Debevec (`www.debevec.org/probes`).*

in the BTF and can be explored by users by changing light and view directions arbitrarily. That this is a real asset of the present approach is also illustrated in figure 5.13, where the defect mentioned above is clearly visible for some light and view direction combinations – and almost invisible for others.

Apparently, the geometric reconstruction using the visual hull algorithm comes nowhere near the spatial precision that a laser range scanner would be able to produce (at least for favourable, well-suited objects). However, numerous applications do not require the full geometric precision deliverable by laser range scanning, in particular demonstration and presentation applications. Although an object's geometry is indispensable even in this type of application for example for shadow computation or interaction, a coarse geometric representation generally suffices, see [Sattler et al. 2005].

To further improve the geometric reconstruction, if desired, any of the numerous extensions of the visual hull reconstruction algorithm as e.g. presented in [Isidoro & Sclaroff 2003], [Slabaugh et al. 2004], [Li et al. 2003a], [Li et al. 2003b], or [Grauman et al. 2003] can naturally be incorporated into this approach. A promising direction for future work is also the incorporation of reconstruction techniques based on photometric correspondences as proposed in [Furukawa et al. 2005].

Compared to laser range scanning, the presented technique facilitates digitisation of a greater scope of small to medium-sized objects in a fast, easy and user-friendly way. Still, in terms of precision, geometry acquisition using laser range scanners is unparalleled. The call for a ready-for-all off-the-shelf data acquisition system is yet unanswered, and the choice of the best-suited acquisition technique generally depends on the object to be digitised and often also on the application at hand.

Part III

Surface Completion

CHAPTER 6

Hole Detection

The conceptual simplicity of point set surfaces makes them suitable for both modelling as well as high quality rendering, even though the original surface is encoded implicitly only in the sampling points. Compared to mesh-based representations, the lack of explicit connectivity information simplifies the definition and implementation of many tasks encountered in geometric modelling, such that for instance free-form deformation techniques for point sets become increasingly popular [Pauly et al. 2003; Botsch & Kobbelt 2005]. On the other hand, the detection of holes in the surface – trivial in the case of meshes – becomes an ill-defined problem.

The knowledge of holes in the data, however, is vital for many applications dealing with point set surfaces and can be exploited in several ways. It can be used to reconstruct surfaces with boundaries or to direct a further scanning step, gathering missing information in holes, either manually or even automatically. In postprocessing, a smoothing step to remove noise profits from boundary information as many smoothing operators usually fail on boundaries and special handling is required at the borders. Identification of points on the boundary of a hole is obviously required before any attempt to algorithmically fill holes, an application useful not only in surface repairing but also in modelling and interactive editing.

While several authors proposed sampling conditions for surfaces to ensure correct reconstruction (most notably [Amenta et al. 1998]), the methods introduced in this chapter are not primarily

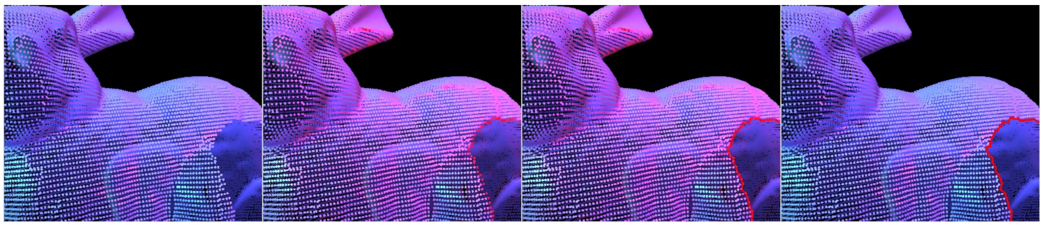


Figure 6.1: *The steps of the boundary detection algorithm. From left to right: A boundary probability* $\Pi(\mathbf{p})$ *is computed for every point (the points are shaded in red according to their boundary probability). Then points are classified into boundary and interior points, exploiting coherence. Finally, for each hole a boundary loop is extracted.*

concerned with undersampling but with holes that a human user might identify when inspecting a point cloud, often unaware of the original surface. Also, as the classification of empty regions in space as *holes in a surface* is arguably application-dependent, the presented methods aim at providing users with intuitive parameters making it easy to find the holes needed for a given application.

6.1 Related Work

The problem of detecting holes in point set surfaces is closely related to surface reconstruction as well as feature extraction. Thus, many algorithms in those areas include criteria to identify holes or undersampled surface patches.

[Gumhold et al. 2001], [Linsen & Prautzsch 2002] as well as [Moenning & Dodgson 2004] apply what shall be referred to as the *angle criterion* for the remainder of this chapter. The angle criterion considers for each sample point $\mathbf{p}$ a set of neighbouring samples and examines the maximum angle between two consecutive neighbours. The idea behind this criterion is that an uneven distribution of neighbours around $\mathbf{p}$ and the resulting large values for the maximum angle indicate the presence of a hole in the surface sampling close to $\mathbf{p}$, or, in other words, a high value for the maximum angle indicates a high probability that $\mathbf{p}$ is on a boundary of the surface sampling.

In addition to this, [Gumhold et al. 2001] also use the correlation matrix formed by the neighbourhood around $\mathbf{p}$. The eigenvectors and eigenvalues of this matrix define a correlation ellipsoid. Its shape, expressed in the ratios of the eigenvalues, is used to identify corner, crease and boundary points and also gives an approximation to crease and boundary direction. In order to find continuous crease lines, a neighbourhood graph on the point set is built and its edges are weighted according to the crease probability. Edges with high probability are then collected and constitute the feature patterns.

In [Dey & Giesen 2001], undersampled regions are detected using the sampling requirement of [Amenta et al. 1998]. This sampling condition is based on an approximation of the medial axis by so called poles of each sample's Voronoi cell. The distance of each point to the medial axis gives the local feature size. Every point on the true surface needs at least one sample point within a ball defined by the local feature size and a factor r. Consequently, the approach of Dey et al. fails to identify holes in flat areas of the surface, where only very few samples are required to fulfill this requirement (in flat areas the medial axis is far away). In these areas, though, often holes exist and are clearly visible for a human observer. Vice versa, most applications do not need to identify "holes" in regions that are declared undersampled at sharp creases, where the sampling requirement can never be met (at sharp edges the medial axis touches the surface, suggesting the need for an infinite sampling density).

6.2 Overview

Let, as usual, $\mathcal{S}$ be a 2-manifold surface and let the set of points $\mathcal{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\} \subset \mathbb{R}^3$ be a (not necessarily regular) sampling of $\mathcal{S}$. Suppose also that $\mathbf{n}_1, \ldots, \mathbf{n}_N$ are the corresponding surface normals. The problem is now to define an operator

$$B_\mathcal{P} : \mathcal{P} \to 2^\mathcal{P} \;; \quad B_\mathcal{P}(\mathcal{P}) \mapsto \{\mathbf{p} \in \mathcal{P} \mid \mathbf{p} \text{ is boundary}\}$$

that identifies the set of boundary points $\mathcal{B} = B_\mathcal{P}(\mathcal{P})$ circumscribing holes in $\mathcal{P}$. The boundary operator is denoted with a subscript $\mathcal{P}$ to stress that the assignment *boundary* or *non-boundary* is strictly a property of the point set under consideration itself.

The basic layout of the hole detection scheme presented here (depicted in figure 6.1) is as follows: For each point $\mathbf{p} \in \mathcal{P}$ a boundary probability $\Pi(\mathbf{p})$ is computed, reflecting the probability that $\mathbf{p}$ is located on or near a hole in the surface sampling (section 6.3). The fact that the boundary property is coherent, i.e. boundary points have proximate neighbours that are also boundary, is exploited thereafter to construct closed loops circumscribing the hole in a shortest cost path manner (section 6.4). Results and applications are then given in section 6.5.

6.3 Boundary Probability

The property of *being boundary* inherently is a property of the local neighbourhood of $\mathbf{p}$ rather than of the point $\mathbf{p}$ itself. In order to define and evaluate the boundary criteria, we therefore have to seize the local neighbourhood $N_{\mathbf{p}}$ more formally.[1]

6.3.1 Neighbourhood Collection

A very common definition of local neighbourhoods around a point $\mathbf{p}$ found in the literature is the k-neighbourhood $N_{\mathbf{p}}^{k}$, consisting of the k nearest samples in $\mathcal{P}$ to $\mathbf{p}$. This simple definition, though, becomes unreliable in areas of varying sampling density, which are of particular interest here. For points lying on the edge between a densely and a sparsely sampled region, the k-neighbourhood will be biased towards the dense region (figure 6.2, left).

This problem can be alleviated to some extent by the $N_{\mathbf{p}}^{k\epsilon}$ neighbourhood, that includes not only the k nearest points but also all points inside a sphere with radius ϵ. By selecting an appropriate value for ϵ, the biasing effect can be reduced, but the

[1]The letter N is used in this chapter in two different meanings: On the one hand $N_{\mathbf{p}}^{k}$, $N_{\mathbf{p}}^{k\epsilon}$, $N_{\mathbf{p}}$, etc. denote neighbourhoods around a certain point $\mathbf{p}$, and on the other hand N (without super- or subscript) denotes the total number of points in the point set. Nevertheless, the notations are used in parallel in this chapter for consistency reasons and should be unambiguous in any case.

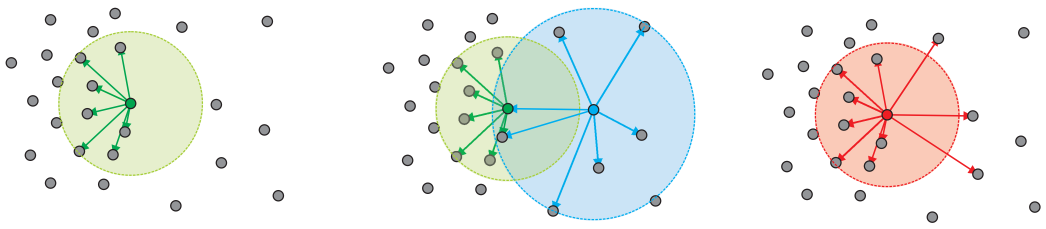


Figure 6.2: *The k-neighbourhood is biased towards densely sampled regions. The neighbourhood of points in the densely sampled region may not contain close-by points of the sparsely sampled area* (left and middle). *The symmetric $k\epsilon$-neighbourhood is not affected in this example by the change in sampling density* (right).

neighbourhood of points in densely sampled regions will contain more points than necessary, increasing the cost of evaluating the boundary criteria, which effectively limits the range of a feasible ϵ. For sharp sampling density changes (as often encountered in point sets stemming from registered range images), this alleviation alone is not sufficient.

Another important aspect is that this neighbourhood relation is not a symmetric one. In particular for points $\mathbf{p}$ situated on a sampling density drop, it may happen that the $k\epsilon$-neighbourhood $N_{\mathbf{p}}^{k\epsilon}$ contains only points in the more densely sampled region. In this specific situation, however, $\mathbf{p}$ is typically contained in the neighbourhood of some nearby points from the more sparsely sampled region (figure 6.2, middle). To overcome the aforementioned biasing effect, it therefore often suffices to include these nearby points in the neighbourhood (figure 6.2, right). To complete the neighbourhood for the critical points, one can hence define the symmetric neighbourhood:

$$N_{\mathbf{p}} = \left\{\mathbf{q} \in \mathcal{P} \mid \mathbf{q} \in N_{\mathbf{p}}^{k\epsilon} \vee \mathbf{p} \in N_{\mathbf{q}}^{k\epsilon}\right\},$$

i.e. $\mathbf{q}$ is considered one of $\mathbf{p}$'s neighbours, already if $\mathbf{p}$ is one of $\mathbf{q}$'s.

For efficient collection of the points contained in the neighbourhood for each point, a kd-tree is built, containing all points in $\mathcal{P}$. The kd-tree supports the collection of the k nearest neighbours to

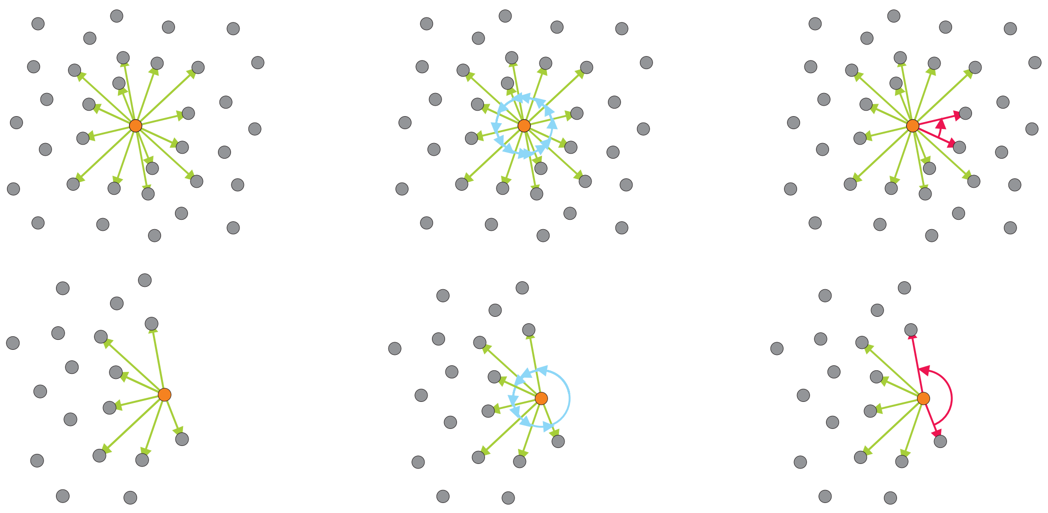


Figure 6.3: *The three steps in the evaluation of the angle criterion for an interior point* (top row) *and a boundary point* (bottom row)*. After projection onto the tangent plane the difference vectors are generated* (left)*. The projected points are sorted into a cyclic ordering around* **p** (middle)*. The largest angular gap between two consecutive points is used to compute the boundary probability* (right)*.*

a point in $O(k \log^3 |\mathcal{P}|)$ and can also be used to quickly retrieve all points in a sphere of radius ϵ. After constructing the kd-tree, the *proximity graph* $G(\mathcal{P}, \mathcal{E})$ is built, with the points in $\mathcal{P}$ as vertices and edges

$$\mathcal{E} = \{(i, j) \mid \mathbf{p}_j \in N_{\mathbf{p}_i}\} .$$

Please note that this graph is symmetric, and the adjacency lists of the graph correspond to the $N_{\mathbf{p}}$-neighbourhood of each point.

6.3.2 The Angle Criterion

The angle criterion projects all neighbouring points contained in $N_{\mathbf{p}}$ onto the tangent plane and sorts the projections into a cyclic ordering around $\mathbf{p}$, see figure 6.3 (middle column). The basic idea is that the largest angular gap g between two consecutive projected neighbours will be significantly larger for a boundary point than for an interior point, as illustrated in figure 6.3 (right column). Hence, one can define as probability measure for a point $\mathbf{p}$ to be situated on a boundary according to the angle criterion:

$$\Pi_{\angle}(\mathbf{p}) = \min\left(\frac{g - \frac{2\pi}{|N_{\mathbf{p}}|}}{\pi - \frac{2\pi}{|N_{\mathbf{p}}|}}, 1\right). \tag{6.1}$$

In this formula, relating $g - 2\pi/|N_{\mathbf{p}}|$ to $\pi - 2\pi/|N_{\mathbf{p}}|$ (instead of relating g to π themselves) results in a more robust probability measure that is less influenced with respect to the number of points in the neighbourhood, and thereby with respect to the overall sampling density, and to the parameters N and ϵ.

In contrast to the angle criterion used in the existing literature, it is advisable to ignore points $\mathbf{q} \in N_{\mathbf{p}}$ with a small scalar product $\langle \mathbf{n}_{\mathbf{p}}, \mathbf{q} - \mathbf{p} \rangle$ during the evaluation of equation (6.1). This way the angle criterion becomes less susceptible to small inaccuracies in the normal direction.

6.3.3 The Half-Disk Criterion

In 2D-image processing, edge detection algorithms in their various forms basically identify pixels whose luminance deviates considerably from the average luminance of their neighbouring pixels. The same rationale can also be applied in the present problem setting. On a 2-manifold, the neighbourhood of points in the interior of the surface is homeomorphic to a disk such that we can expect the difference between the point $\mathbf{p}$ itself and the *average* – represented by the centroid (centre of mass) of its neighbourhood – to be small and non-zero mainly in direction of the surface normal. On the contrary for points on a boundary: Their neighbourhood is shaped like a half-disk, whose centroid significantly deviates from $\mathbf{p}$ (see figure 6.4).

Fortunately, this fundamental distinction prevails not only in the continuous setting but also in case of regular surface samplings, and to a lesser extent even for irregular surface samplings. As a consequence, the probability of a point $\mathbf{p}$ to be boundary can be expected to be high where the distance between $\mathbf{p}$ and the

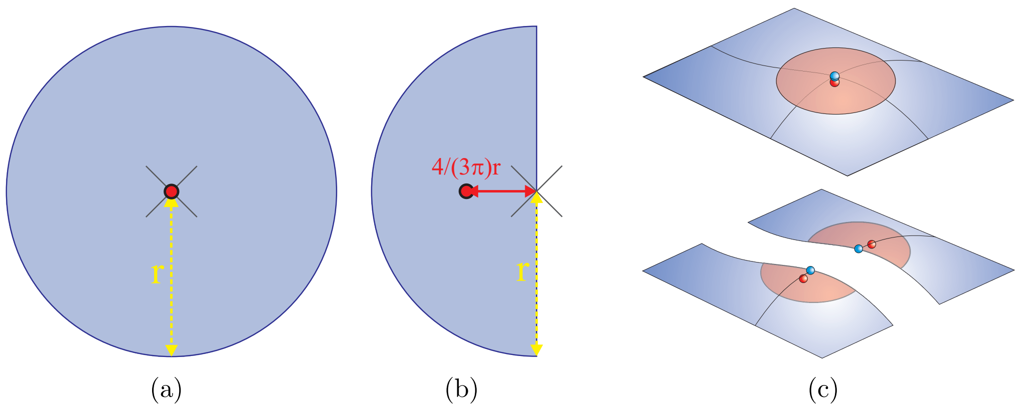


Figure 6.4: (a) *The local neighbourhood of points located in the interior of the surface is essentially shaped as a disk, such that the average of the neighbourhood points will coincide with the interior point.* (b) *In contrast, for points located on the boundary of a surface it will deviate in direction of the interior surface.* (c) *For non-flat surfaces, there is also a deviation into the direction of the curvature of the surface, which is to be ignored for the purpose of boundary detection.*

centroid of its neighbourhood is large. This geometric observation leads to the so-called *Half-Disk Criterion* $\Pi_\mu(\mathbf{p})$.

Let $\mu_\mathbf{p}$ be the weighted average of all neighbours of $\mathbf{p}$:

$$\mu_\mathbf{p} = \frac{\sum_{\mathbf{q}\in N_\mathbf{p}} g_\sigma(\|\mathbf{q}-\mathbf{p}\|)\mathbf{q}}{\sum_{\mathbf{q}\in N_\mathbf{p}} g_\sigma(\|\mathbf{q}-\mathbf{p}\|)},$$

where $g_\sigma(d) = \exp\left(-d^2/\sigma^2\right)$. Including these weights into the average computation has two beneficial effects in comparison to the standard centroid formulation: On the one hand, it reduces the influence of variations in the sampling density. On the other hand and more importantly, it counteracts the adverse effect that more distant neighbours receive a bigger weight than closer neighbours in the standard centroid computation and only slightly differing distributions of sample points may therefore have significant effect on the computed centroid.

The gaussian parameter σ depends on the average distance to the neighbouring points $r_\mathbf{p}$ and is set to

$$\sigma = \frac{1}{3} r_\mathbf{p}$$

such that the influence of points outside the neighbourhood $N_\mathbf{p}$ can be neglected.

As final modification to the standard centroid computation, the centroid $\mu_\mathbf{p}$ is projected onto the tangent plane. This way, deviations in direction of the surface normal (e.g. resulting from a significant curvature of the surface) have no influence on the centroid.

Collecting the pieces, this delivers:

$$\Pi_\mu(\mathbf{p}) = \min\left(\frac{\|\mathbf{p} - (\overline{\mu}_\mathbf{p})\|}{\frac{4}{3\pi} r_\mathbf{p}}\ ,\ 1\right),$$

where $\overline{\mu}_\mathbf{p}$ is the projection of $\mu_\mathbf{p}$ onto the tangent plane. The reference quantity $\frac{4}{3\pi} r_\mathbf{p}$ is the distance of the centroids of a half-disk and a full disk, respectively.

6.3.4 The Shape Criterion

As known from statistical analysis and as noted in [Gumhold et al. 2001], the overall distribution of the neighbouring points $N_\mathbf{p}$ is captured in the shape of the so-called *correlation ellipsoid* (see figure 6.5). This ellipsoid is defined by the eigenvalues $\lambda_0 \geq \lambda_1 \geq \lambda_2$ and the corresponding eigenvectors of the weighted covariance matrix $C_\mathbf{p}$:

$$C_\mathbf{p} = \sum_{\mathbf{q} \in N_\mathbf{p}} w(\mathbf{q})(\mu_\mathbf{p} - \mathbf{q})(\mu_\mathbf{p} - \mathbf{q})^t. \tag{6.2}$$

It is therefore possible to exploit the relative magnitudes of the eigenvalues to derive a further probability measure for a point being on the boundary. Hence, the three relative magnitudes are collected in a decision vector

$$\Lambda_\mathbf{p} = (\bar{\lambda}_0, \bar{\lambda}_1, \bar{\lambda}_2),$$

with $\bar{\lambda}_i = \lambda_i/(\lambda_0+\lambda_1+\lambda_2)$. There are four characteristic situations $\phi \in \Phi = \{\mathit{Boundary}, \mathit{Interior}, \mathit{Corner/Noise}, \mathit{Line}\}$, each with a representative decision vector:

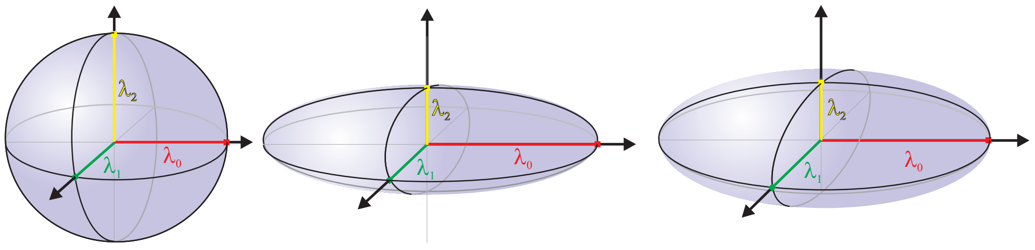


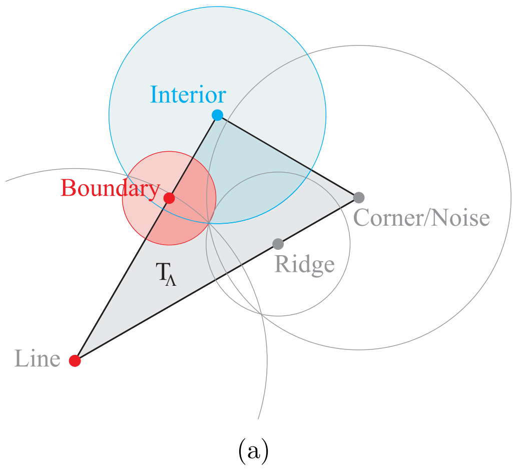
Figure 6.5: *The distribution of points in $N_{\mathbf{p}}$ can be analysed using statistical methods to compute a correlation ellipsoid. The lengths of the ellipsoid's main axes correspond to the eigenvalues $\lambda_0 \geq \lambda_1 \geq \lambda_2$ of $N_{\mathbf{p}}$'s covariance matrix. For a point in the interior of a point sampling of a 2-manifold surface, we would have $\lambda_0 \approx \lambda_1 \gg \lambda_2$* (right)*, whereas a point in on the boundary of such a sampling we would rather have $\lambda_0 \approx 2\lambda_1 \gg \lambda_2$* (middle)*. The case $\lambda_0 \approx \lambda_1 \approx \lambda_2$ indicates a more or less even distribution of points around* $\mathbf{p}$ *such that not even a confident normal for the approximated surface can be estimated* (left).

$\phi =$ Boundary	$\Lambda_\phi = (\frac{2}{3}, \frac{1}{3}, 0)$
$\phi =$ Interior	$\Lambda_\phi = (\frac{1}{2}, \frac{1}{2}, 0)$
$\phi =$ Corner/Noise	$\Lambda_\phi = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$
$\phi =$ Line	$\Lambda_\phi = (1, 0, 0)$

The basic idea is now to compute a decision vector for each point in the point set and to compute a boundary probability for this point based on the location of its decision vector with respect to the above shape representatives.

The last three values for Λ given in the table above span the decision triangle T_Λ depicted in figure 6.6. Please note that this triangle contains all possible Λ values, because the covariance matrix defined in (6.2) is positive semi-definite.

Tentative classification probabilities $\widetilde{\Pi}_\phi$ can now be extracted for each of the situations described above from $\Lambda_{\mathbf{p}}$ by evaluating a spatial kernel around the characteristic Λ_ϕ. Like in the previous section, a Gauss kernel g_σ with $\sigma_\phi = \frac{1}{3}\|\Lambda_\phi - \mathit{centroid}(T_\Lambda)\|^2$ is used, effectively defining a radius of influence for each of the characteristic points (see figure 6.6, left). Now $\widetilde{\Pi}_\phi$ is for each shape



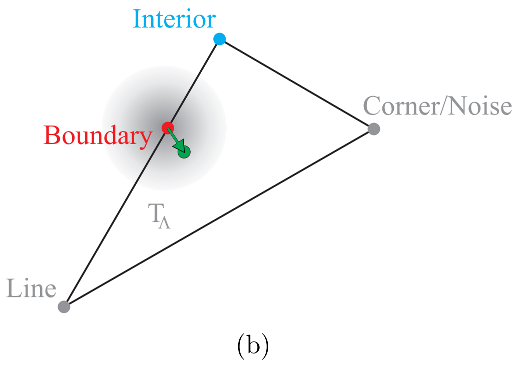


Figure 6.6: (a) *The triangle spanned by the representative Λ_Φ-vectors with highlighted characteristic points for certain shapes Φ. The circles passing through the triangle centroid* **c** *are shown for every shape in the respective colour.* (b) *The tentative probability $\widetilde{\Pi}_{boundary}$ is computed by evaluating the kernel around the characteristic Λ for boundary points.*

$\phi \in \Phi$ given as

$$\widetilde{\Pi}_\phi(\mathbf{p}) = g_{\sigma_\phi}(\|\Lambda_\mathbf{p} - \Lambda_\phi\|)$$

Obviously, the regions for different shapes overlap. Normalisation leads to

$$\Pi_\phi(\mathbf{p}) = \frac{\widetilde{\Pi}_\phi(\mathbf{p})}{\sum_{\varphi\in\Phi} \widetilde{\Pi}_\varphi(\mathbf{p})}. \tag{6.3}$$

6.3.5 Combining the Criteria

Every one of the three above criteria has its own advantages. Compared to the angle criterion, the half-disk criterion is better capable of detecting small holes, especially when the hole is crossed by some edges of the neighbourhood graph, see figure 6.7.

On the other hand, while the half-disk and the ellipsoid criterion typically find narrow bands of boundary points around holes (in particular for larger k) the angle criterion is sharper and better exposes thin lines of boundary points (see figure 6.8). In the presence of noise, finally, the shape criterion performs best (see figure 6.9).

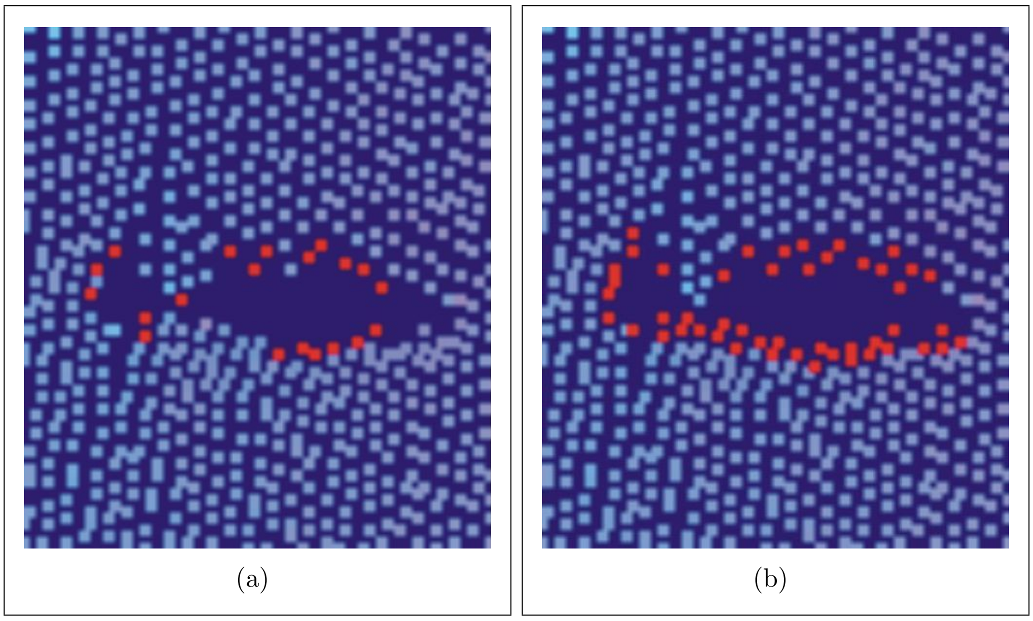


Figure 6.7: *A small hole, crossed by some edges of the neighbourhood graph G. Points with a boundary probability as computed with the angle criterion* (a) *and the half-disk criterion* (b) *above a threshold of* 0.5 *are highlighted in red colour.*

In order to make use of the different capabilities of the criteria and to increase the robustness of the boundary probability computation, we derive a combined boundary probability into a weighted sum

$$\Pi(\mathbf{p}) = w_\angle \Pi_\angle(\mathbf{p}) + w_\mu \Pi_\mu(\mathbf{p}) + w_\phi \Pi_{\phi = \text{ Boundary}}(\mathbf{p}). \qquad (6.4)$$

The weights $w_\angle$, w_μ and w_ϕ, where $w_\angle + w_\mu + w_\phi = 1$, can be adjusted by the user upon visual inspection. As default, a uniform weighting scheme produces good results, but for noisy models, the weight of the shape criterion should be increased.

6.3.6 Normal Estimation

By construction, both, the angle and the average criterion, depend heavily on the normal at the point $\mathbf{p}$. Therefore, a good estimation of the normal is mandatory in cases where no normal information is available otherwise. Following the normal estimation method

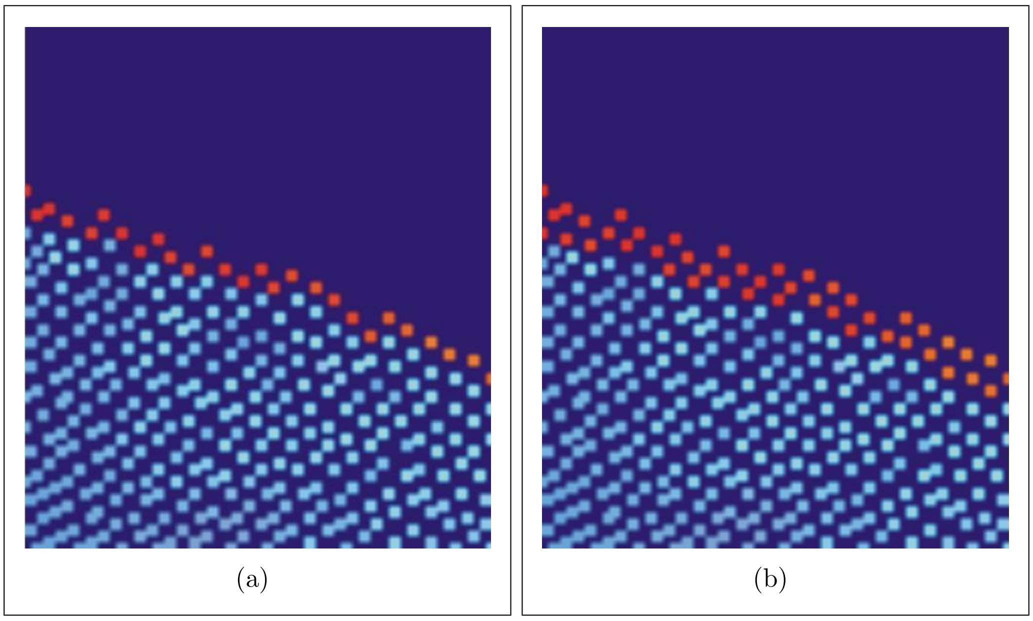


Figure 6.8: *Boundary points detected by the half-disk criterion form a band of boundary points* (b)*, whereas the angle criterion finds a sharp boundary* (a)*.*

by Hoppe et al. [1992], the normal can be estimated as the eigenvector corresponding to the smallest eigenvalue of the weighted covariance matrix of $N_{\mathbf{p}}$ (eqn. (6.2)).

For some configurations, however, this normal estimation fails. In particular at sharp creases, the best fitting plane as defined by the smallest eigenvalue might be indeed orthogonal to the underlying surface (see figure 6.10). Fortunately, it is possible to exploit information gathered during the computation of the angle criterion to detect such cases in the normal estimation process [Linsen & Prautzsch 2002].

To this end, the angle criterion is evaluated after the normal has been estimated. If the boundary probability $\Pi_{\angle}(\mathbf{p})$ exceeds a given threshold, the estimated normal is rotated by 90 degrees about the axis defined by the two points on both sides of the maximum gap, projected into the tangent plane. Then the angle criterion is evaluated again, this time using the rotated normal, and the new normal is kept if the boundary probability has been reduced significantly, i.e. by more than 50%.

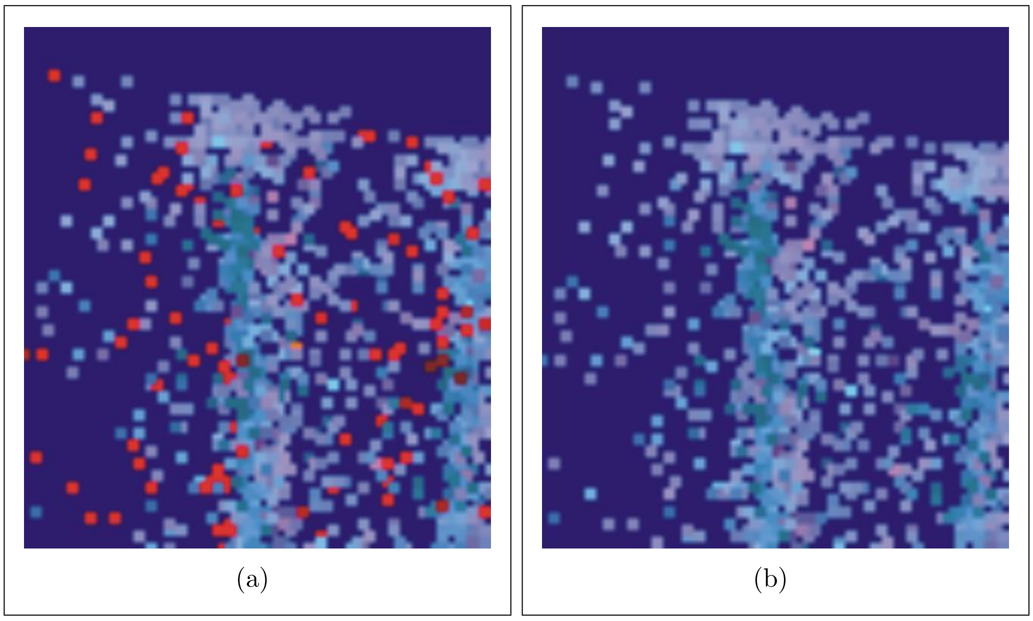


Figure 6.9: *The angle criterion* (a) *identifies many false boundary points in the presence of noise, while the shape criterion* (b) *is not affected.*

Please note that a consistent orientation of the estimated normals is not required for any of our criteria. Nevertheless, if required for other purposes (e.g. visualisation), it can easily be achieved by applying the minimum spanning tree traversal introduced in [Hoppe et al. 1992] on the neighbourhood graph.

6.4 Boundary Loops

The extraction stage of the boundary detection algorithm aims at producing a classification for each point, stating if it is a boundary or an interior point. In addition to the boundary probability computed with the scheme described in the last section, the coherence between boundary points will be exploited for the classification. This greatly improves the robustness of the presented method. Moreover, connected loops of points, circumscribing a hole, will be found, providing immediate access to the boundary.

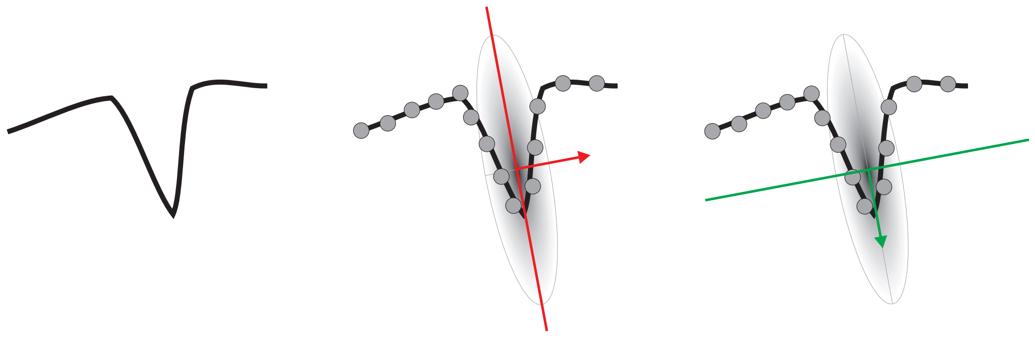


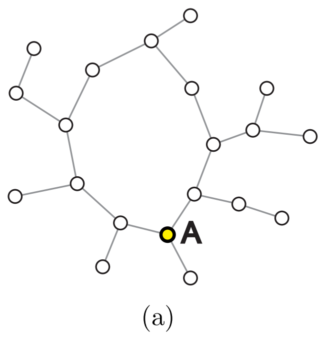
Figure 6.10: *In sharp creases the best fitting plane (illustrated with the correlation ellipsoid in the middle) might be orthogonal to the underlying surface* (left)*. These cases can be detected with the angle criterion, and the normal can then be flipped accordingly* (right)*.*

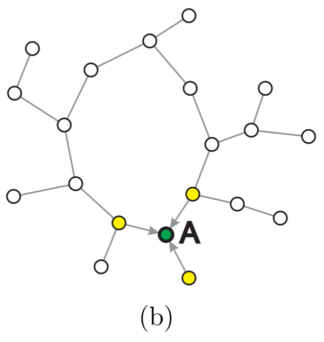
6.4.1 Boundary Coherence

Any point on a boundary loop has at least one neighbour to each side also belonging to the boundary. This property can easily be exploited using an iterative classification procedure. First, all points with a boundary probability greater than a user-defined threshold are declared boundary points. Then, for each of these points, the two neighbours forming the maximum gap in the sense of the angle criterion are found. A point stays classified as boundary point if and only if both of these neighbouring points have also been declared boundary points. All other points are marked as interior points. This process is repeated until no more points change their status. Note that only the neighbours of points that did change status in the previous iteration have to be reconsidered in the following step, making the classification very efficient.

After the classification, we use an algorithm that is built upon the one presented in [Gumhold et al. 2001] to construct a minimum spanning graph (MSG) based on the proximity graph G. By construction, this MSG will contain loops if and only if they correspond to the boundary loops we are interested in.

To this end, we use an extension of Kruskal's minimum spanning tree algorithm. The required *edge weights* $w(i, j)$, are derived similarly to [Gumhold et al. 2001] in two parts. The first compo-





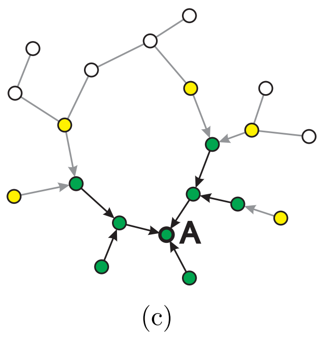


Figure 6.11: *The first three steps in the loop extraction:* (a) *The breadth first graph traversal is spawned at vertex **A**.* (b) ***A** is marked green, all adjacent vertices are queued for visitation and marked yellow. The arrows indicate the vertices' predecessors.* (c) *Visited vertices are marked green, newly discovered vertices are queued for visitation.*

nent penalises the boundary probability of the adjacent points:

$$w_{\text{prob}}(i,j) = 2 - \Pi(\mathbf{p}_i) - \Pi(\mathbf{p}_j).$$

The second component incorporates the local sampling density measured by $r_{\mathbf{p}}$ (defined as the average distance to $\mathbf{p}$'s neighbours (see section 6.3.3) and penalises long edges so that the boundary loops will contain as many boundary points as possible:

$$w_{\text{density}}(i,j) = \frac{2\|\mathbf{p}_i - \mathbf{p}_j\|}{r_{\mathbf{p}_i} + r_{\mathbf{p}_j}}.$$

The total weight is then given by

$$w_{\text{total}}(i,j) = w_{\text{prob}}(i,j) + w_{\text{density}}(i,j).$$

With these weights, the algorithm is then initiated as follows: In the beginning, every vertex of G constitutes a stand-alone component in G. Then all eligible edges are processed in ascending order, according to their weight. Here, an edge (i,j) is considered eligible only if $w_{\text{prob}}(i,j)$ and $w_{\text{total}}(i,j)$ are below pre-defined thresholds. A threshold combination of 1.1 and 3 proved good in practice and was used for all the examples given.

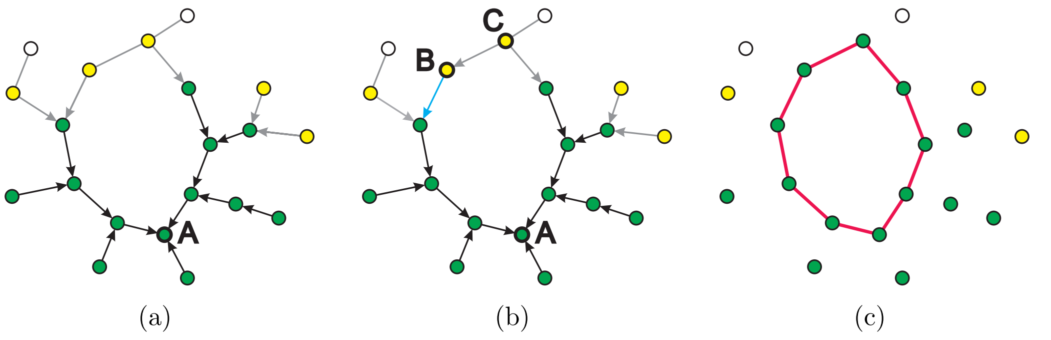


Figure 6.12: *Loop extraction (contd.)* (a) *One further step in the graph traversal.* (b) *In this step, when **B** is visited and to be marked green, it is discovered that one of its adjacent vertices (**C**) is already marked yellow. A loop has been found and can be extracted by tracing back the predecessors of both vertices.* (c) *The extracted loop.*

If an edge (i, j) connects two distinct components of G, the edge is added to the MSG and the two components are joined. If, on the other hand, the edge connects two vertices of the same component, it is included in the MSG only if the emerging loop is longer than a predefined minimum loop length e, measured as the number of edges in the loop. Similar to the radius ϵ in the construction of the neighbourhood graph G, the minimum loop length e steers the minimal hole radius to be found. Although independent in theory, these two parameters are therefore semantically correlated and one can define $e \propto \frac{2\pi\epsilon}{d}$, where d is the average edge length in the graph.

6.4.2 Loop Extraction

With the MSG at hand, the boundary loops can be extracted using a breadth first search. The search is started at an arbitrary vertex in the MSG. The algorithm maintains for all vertices a colour value signaling one of three states: white (untouched vertices), yellow (queued for visitation) or green (already processed). In the beginning, all vertices are white, except the origin, which is yellow (see figure 6.11). In the following steps, vertices on the front of the queue are removed from the queue and marked green. Its adjacent vertices are treated in dependence of their colour (figures 6.11 and 6.12):

- **White** vertices are inserted into the queue and marked yellow.
- **Green** vertices are ignored.
- If one of the adjacent vertices is **yellow**, a loop has been found and the traversal is stopped.

After the traversal is stopped, the loop can be extracted by tracing back the two vertices that caused the traversal to stop (vertices **B** and **C** in figure 6.12).

After all vertices belonging to the loop are removed from the MSG, the search is re-launched with an arbitrary vertex from the remaining MSG and iterated until no more loops can be found.

In a final step, all points belonging to a loop are marked as boundary points.

6.5 Results and Conclusions

The presented algorithm was applied to a variety of models. Figure 6.13 illustrates the effect of the symmetric neighbourhood graph that is designed particularly to filter out even abrupt sampling density changes, a situation which causes even well-established hole criteria to fail. For this example, one half of the depicted data set was heavily down-sampled and only the angle criterion employed. Note how well the drastic change in sampling density is handled.

Nevertheless, the robust detection of holes in the presence of noise or of holes of small size remains a challenge using only this criterion. To overcome this, two novel boundary criteria have been introduced: The half-disk criterion is the 3D-analogue to the well-known border detection in images, whereas the shape criterion exploits a classification scheme based on local data analysis.

The notion of a hole is inherently and per-se ill-defined in the context of point set surfaces, and hence any classification ultimately needs to adapt to the application's (or rather the user's)

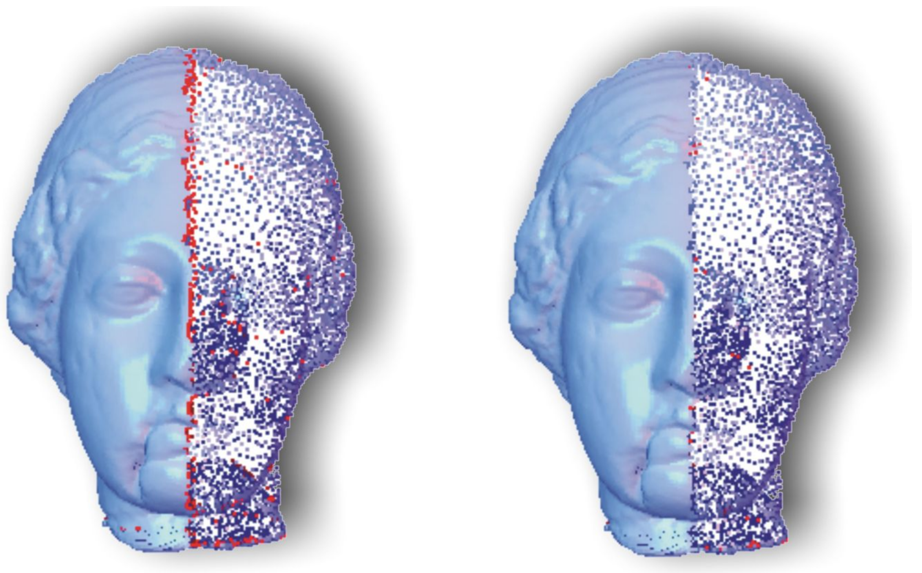


Figure 6.13: *The effect of the symmetric neighbourhood relation. Left: k-nearest neighbours Right: Symmetric neighbourhood graph*

interpretation. Consequently, the presented probabilistic approach can be trimmed using intuitive parameters, rendering the method easily adjustable to the task at hand. The parameter k of the neighbourhood definition determines the size of the local neighbourhoods. If k is increased, only larger holes can be detected, as smaller holes will be crossed by edges of G. For the examples depicted in this (and the following) chapter, k was set to a value between 12 and 25, depending on the amount of noise present in the data. If there is considerable noise, larger values of k can be used to improve the robustness of the hole detection, while the parameter ϵ can be used to define a minimum hole size, since the neighbourhood will stretch over all holes with a diameter less than ϵ. This way the user is enabled to focus on the important holes, for instance in the dragon data set, as demonstrated in figure 6.15.

By making use of the coherence between boundary samples, the robustness of the hole detection is further increased. As a by-product of this stage, boundary loops are extracted, delivering subsequent processes direct access to the contours of the holes.

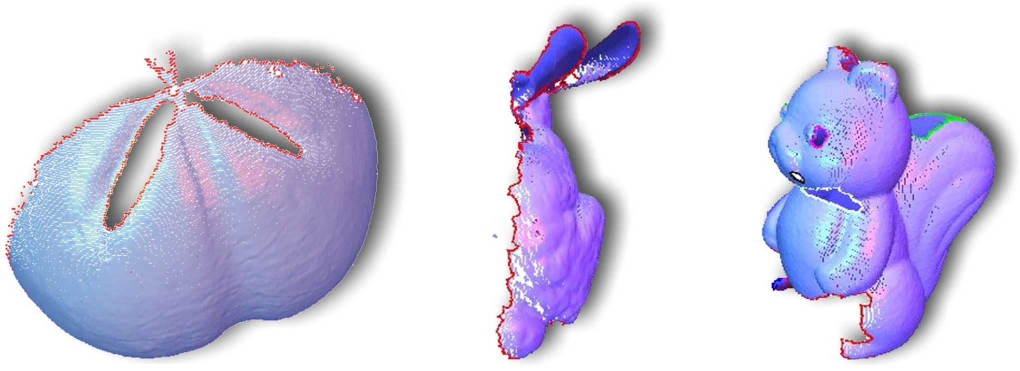


Figure 6.14: *Detected boundaries in a single scan of an echinite, the bunny, and in the registered, yet incomplete squirrel data set. All three criteria were combined with equal weight.*

Figure 6.16 once again illustrates the whole process, from the computation of the boundary probability, over thresholding to the loop extraction case. This example emphasises the ability of the presented algorithm to ignore drastic sampling density drops in most cases. In this data set generated from a reconstructed mesh, the eye region is densely sampled, whereas the surrounding region is sparse. It is interesting to compare these results to the corresponding images in [Dey & Giesen 2001]. According to the criteria proposed by Dey et al., no undersampling occurred in this region of the mannequin head model, a hypothesis that is supported by the presented algorithm with the exception of the small hole depicted in figure 6.16, bottom right. This example also demonstrates that the existence of a hole is often arguable, and that the final classification can ultimately be performed only by the application or by the user. Nevertheless, the figures given in this chapter make evident that, with a little parameter tuning, the presented algorithm is capable of detecting holes in point sets and filtering out undesired or insignificant ones, thereby reducing the number of misidentified cases that would require manual deselection or other postprocessing.

For many applications, such as automatic hole filling, which will be subject of the following chapter, the detection of holes has

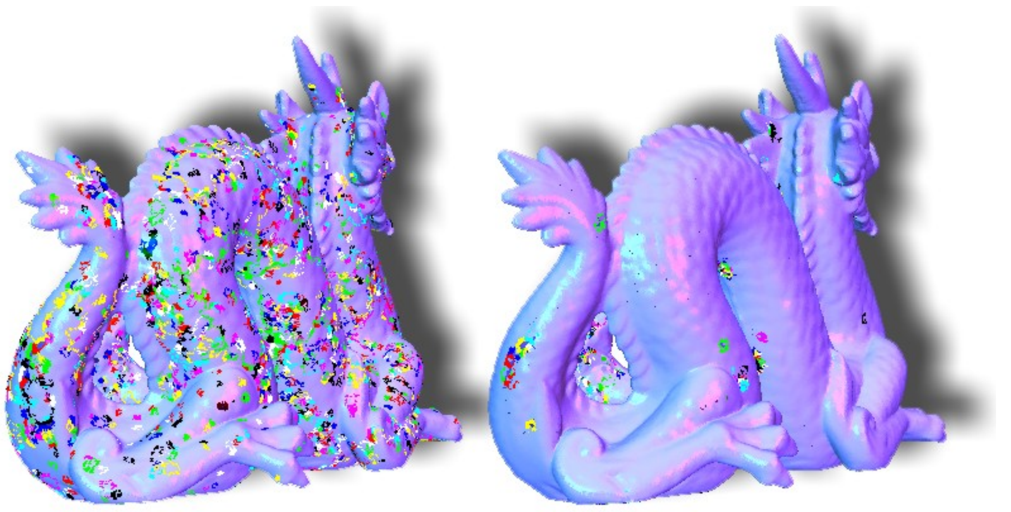


Figure 6.15: *Numerous small holes are detected in the dragon model for* $k = 15$*, but larger holes can be isolated if all points within* 0.01 *of the bounding box diagonal are also included in the neighbourhood.*

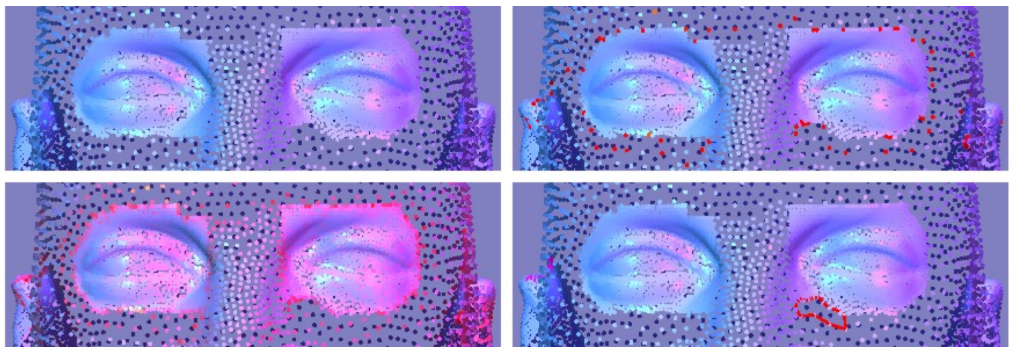


Figure 6.16: *The eye region of the mannequin head model (top left), the boundary probability (bottom left), after thresholding (top right), and after boundary loop extraction (bottom right).*

to be repeated after filling part of the hole. A reasonable efficiency of the hole detection is therefore desirable. In the dragon example (containing over 400000 points) the holes depicted in figure 6.15 (right) were detected in less than two minutes on a AMD Athlon 2.21 Ghz. Specifically, the timings were: Construction of the kd-tree and the symmetrised proximity graph $9s$, computation of the integrated boundary probability $11s$, extraction of the boundary loops $26s$. In the context of hole filling, the update of the boundary loops can naturally be performed incrementally, such that here timings are even considerably faster.

Figure 6.14 finally shows some more examples where the hole detection algorithm correctly identified the boundaries (and holes) in some objects directly stemming from range data, in particular in a single scan of a echinite (already seen in chapter 5), and in a single scan of the bunny. The image to the right shows the already registered, yet incomplete squirrel model. For the bunny, a minimum loop size of $e = 1000$ was used to suppress the detection of loops around the smaller holes.

Chapter 7

Surface Inpainting

In the context of image and art restoration, *Inpainting* is a well-known technique where paint losses – caused by abrasion, destruction, or created during the process of restoration itself – are filled with plaster up to the level of the surrounding paint and then textured and coloured to match. Recent years have seen the establishment of this term also in the processing of digital data, in particular in the field of image completion and later also in video completion (see references below). Analogue tasks, however, can also be found in 3D geometry processing, since digital representations of real-world objects often contain holes, due to hindrances during data acquisition or as a consequence of interactive modelling operations.

Although the creation of digital 3D copies of real-life objects is becoming a standard procedure for numerous application fields and despite all technological progress, models resulting even from the most careful acquisition process are generally incomplete, due to occlusion, the object's material properties or spatial constraints during recording (among others), i.e. they contain under-sampled regions and/or holes. In some applications, holes are also deliberately introduced into an object, since removing damaged, undesired or unnecessary parts of an object is an important tool in interactive modelling.

Various algorithms exist that allow the completion of such holes in a smooth way, but they generally do not lead to visually appealing solutions. To this end, the holes have to be filled *plausibly*, i.e. the basic geometry has to be smoothly patched and the (unknown, yet assumed) detail geometry has to be restored or extrapolated, possibly taking into account the context of the object. That this ill-posed task has hope of being solved at all is based on the observation that real-life objects often exhibit a high degree of coherence in the sense that for missing parts one can find similar regions on the object to draw information from. This observation has been exploited extensively in the field of 2D texture synthesis and image completion, even though real symmetry (or close-to-symmetry) is far less frequently found in photographs (which constitute a 2D projection of a 3D scene) than it is for the 3D objects themselves.

Symmetry and coherence in 3D objects have also been exploited for 3D surface completion. The problem with previous approaches, though, is that they require specific spatial structures to identify holes and potential candidate fragments to be copied to defective regions. Consequently, the results depend heavily on the choice and location of these auxiliary structures. In contrast, this chapter introduces a surface inpainting method that analyses the neighbourhood of a hole, finds best matching *local* neighbourhood patches represented in local frames (the 3D analogue to what is called a *fragment* in image processing), and fills the hole with copies of these. By finding best matches hierarchically on several scales, the hole is filled in conformance with the context with respect to all considered scales.

The key challenges for such a 3D fragment-based approach are

- to identify symmetry, similarity and coherence relationships in the scene or on the object,
- to do this independently of a specific choice of any particular coordinate system or parametrisation, and
- to exploit these relationships to inpaint missing surface regions.

Before a detailed description of how these challenges can be tackled, the following section will shortly review the relevant literature, covering the inspiring works on 2D image processing, but also previous approaches to automatic hole filling for boundary representations in 3D. The novel algorithm is then described first in a one-level, non-hierarchical way in section 7.4, and extended to exploit Guidance Surfaces in a multi-scale approach in section 7.5. This section also introduces the two-layer descriptor required to transfer coarse scale geometry information to finer scale filling operations. A collection of exemplar results produced with the novel approach are presented and discussed in section 7.6.

7.1 2D Approaches

7.1.1 Texture Synthesis

In image processing, synthesising images or parts thereof, comes in two flavours: *Texture Synthesis* and *Image Restoration*. In texture synthesis, from a sample image a new (generally larger) image is to be created that appears realistic and preserves visual properties of the sample image. In the extensive literature, this problem has been approached by explicitly modelling the distribution of the textural features which humans perceive as a specific type of texture [Perlin 1985; Turk 2001], by histogram matching [Heeger & Bergen 1995; Bonet 1997; Portilla & Simoncelli 2000], or Markov random fields [Zhu et al. 1998]. Despite the appealing mathematical formulation, these parametric approaches have been outperformed by non-parametric models that synthesise textures exemplar-based by transferring pixels [Efros & Leung 1999] or patches [Wei & Levoy 2000; Nealen & Alexa 2003] with appropriate neighbourhoods to the new image.

7.1.2 Image Restoration

Image completion aims at filling-in holes in an image that are generated erasing defective, damaged or undesired parts of an image, by extending information available in the remaining image. One obvious but important difference to most texture synthesis approaches is that the undamaged region of the image must remain unmodified, whereas the damaged or missing region has to be filled in. It is due to this parallelism to the classical inpainting

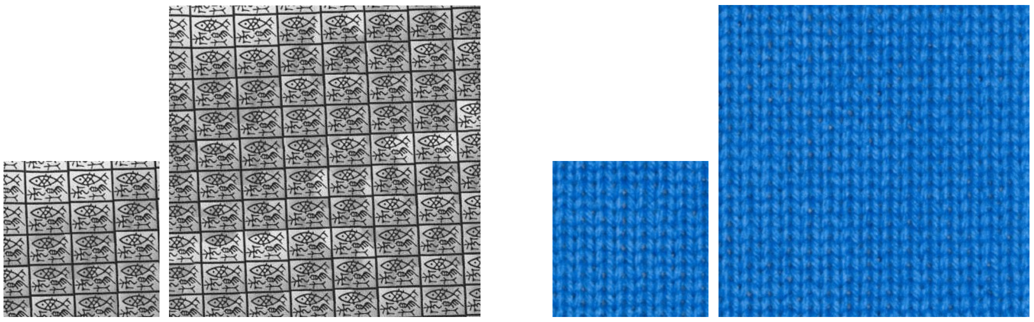


Figure 7.1: *Texture synthesis typically takes a small input image (small images above) and generates an arbitrarily large output image (large images above). Leftmost image taken from [Lin et al. 2004], remaining images courtesy of the computer graphics group in Bonn, see e.g. [Nicoll et al. 2005].*

technique in art restoration that this process is often referred to as *digital image inpainting.*

Furthermore, in addition to the overall visual properties of the image, which are in focus in texture synthesis, the larger and highly irregular structures of the image have to be preserved. With this requirement in mind, Ballester et al. [2001] fill images by explicitly propagating lines of equal brightness (so-called isophotes) by solving variational problems, whereas Jia et al. [2003] segment the image and propagate segment borders into the hole region. Both approaches then fill-in the emerging empty regions in a pixel-based approach. Using isophote-propagation to guide what is otherwise a pure texture synthesis approach, Criminisi et al. [2003] presented an approach that is similar to the one presented here in the sense that the hole filling steps are also *prioritised* in order to propagate larger scale structures to hole regions. To determine the priority criterion, however, feature lines on the surface – lines that can be considered as the 3D analogue to isophotes in images – have to be detected first. The approach presented here also benefits from work presented by Drori et al. [2003], who assign iteratively updated confidence values to each pixel in the image and exploit these confidence values for guiding the filling steps.

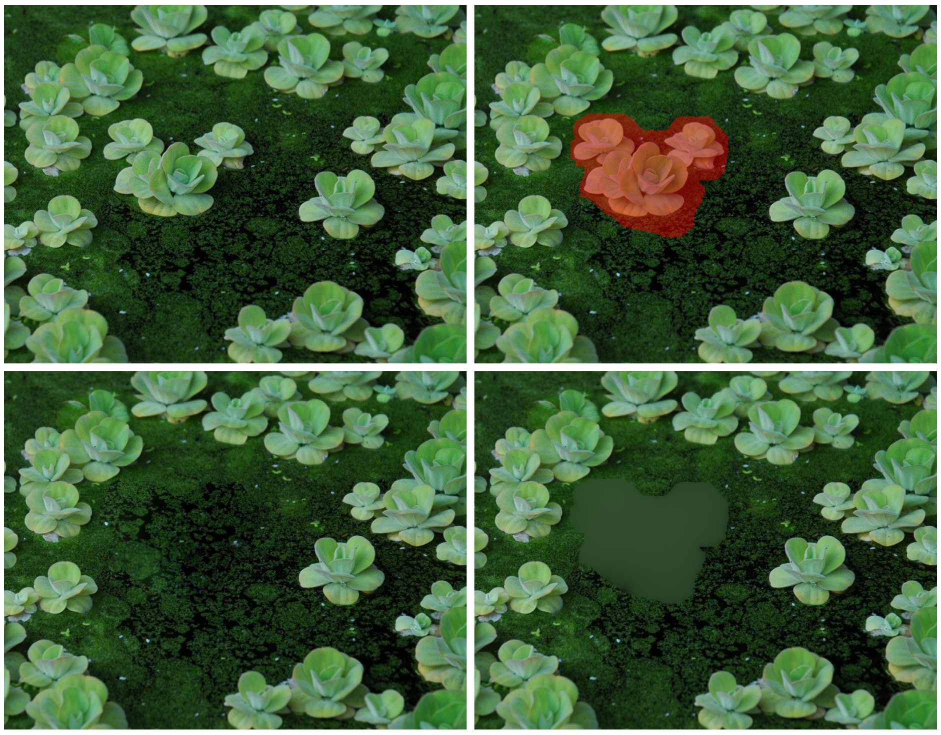


Figure 7.2: *Image completion targets the repair of damaged or undesired image regions. Undesired regions are marked (top right), erased, and automatically filled. The bottom row contrasts a successful with an unsuccessful completion attempt acquired with commercial image processing software.*

$l = 0, \ldots, L$	Hierarchy level (L for coarsest)
$\mathcal{P}^l = \{\mathbf{p}_i^l\}$	Point sets (hierarchy level l)
$\mathcal{B}^l = \{\mathbf{b}_i^l\} \subset \mathcal{P}^l$	Sets of border points
$\mathcal{C}^l = \{\mathbf{c}_i^l\} \subset \mathcal{P}^l \setminus \mathcal{B}^l$	Candidate sets
$\alpha : \mathcal{P}^l \mapsto [0, 1]$	Confidence value
$\mathcal{N}(\mathbf{p}_i^l) \subset \mathcal{P}^l$	Local neighbourhood of p
$\mathcal{G}^l$	Guidance surface
N, N^l	Number of points in $\mathcal{P}^l$
D	Number of points in the descriptor
$\chi : \mathcal{P}^l \to \mathbb{R}^D$	Descriptor

Table 7.1: *Notation and Symbols*

7.2 3D Approaches

As 3D data-acquisition generally leads to incomplete surfaces, filling holes in 3D surfaces is traditionally part of surface reconstruction algorithms (see [Curless & Levoy 1996] as an example), but has also achieved recent research attention in its own right [Davis et al. 2002; Verdera et al. 2003; Liepa 2003; Clarenz et al. 2004].

Lifting the 2D-surface into a 3D volumetric representation, Davis et al. [2002] extend a signed distance function that is initially only defined close to the known surface to the complete space using volumetric diffusion, thereby completing the surface even for non-trivial hole boundaries. Clarenz et al. [2004] cover surface holes minimising Willmore energy functionals, leading to smooth surface patches with guaranteed continuity properties. Various algorithms explicitly refrain to distinguish between holes in the surface and the likewise empty space between sample points; as in reconstruction, these approaches typically generate an implicit representation of the surface first by interpolating e.g. the signed distance field using radial basis functions [Carr et al. 2001][2003]. See also [Steinke et al. 2005] for an example of this type of reconstruction, which relies on a support vector machine-like approach to find the optimal implicit function.

Smooth completion is also the result of very recently published hole filling algorithms, where templates – constructed from some known base mesh [Kraevoy & Sheffer 2005] or from a partly manual selection from a shape data base [Pauly et al. 2005] – have been exploited.

In some applications, however, smooth filling of holes is not sufficient. This is particularly the case in cultural heritage applications, where in addition to an accurate documentation of the status quo of a cultural heritage object, one often requires also visually appealing reconstructions, e.g. for virtual exhibitions etc. In such applications, so-called surface inpainting algorithms are needed that do not only reconstruct the basic geometry of the defective object, but also their fine scale geometric features.

Although the problem of completing 2D surfaces in 3D appears conceptually almost identical to completing 2D images, transferring successful techniques from image completion to 3D is far from trivial. The lack of a regular grid deprives us from the universal parameter domain that is so extensively exploited in 2D image processing. As a consequence, already the construction of multi-scale hierarchies representing different frequency bands – apparently a key ingredient to many image completion approaches – proves to be challenging, as the vertices' positions at the same time encode both, signal *and* domain of the function to be analysed [Taubin 1995; Guskov et al. 1999; Pauly & Gross 2001].

There are yet only few publications that address the problem of detail preservation during hole filling [Savchenko & Kojekine 2002; Sharf et al. 2004]. Adapting technologies from exemplar-based image synthesis methods and similar in concept to our approach, Sharf et al. [2004] fill holes by copying existing surface patches from the object to the hole region. The fundamental problem of this algorithm is that it is completely octree-based: Holes in the surface are detected by checking for near-empty octree cells, different scales in their hierarchical approach are represented through octree levels, descriptors are based on a regular sampling of a distance field, and, most importantly, patches to be copied can be generated from other octree cells only. As a consequence, even perfectly symmetrical objects can only be reconstructed if the symmetry axis of a symmetrical feature happens to coincide with an octree axis (or one of the considered, discrete rotations thereof). Furthermore, due to the resulting non-invariance with respect to rotation, translation and scaling, very careful parameter tuning is required to successfully reconstruct real-world examples.

7.3 Overview and Terminology

Given a point set $\mathcal{P} \subset \mathbb{R}^3$ representing a manifold surface, the goal of our algorithm is to fill any existing holes plausibly, i.e. taking into account the object's local and global context and reconstructing also the fine surface detail that can visually be expected in the missing regions. This goal is achieved by first identifying *Boundary Points*, i.e. points that are close to regions in the point set

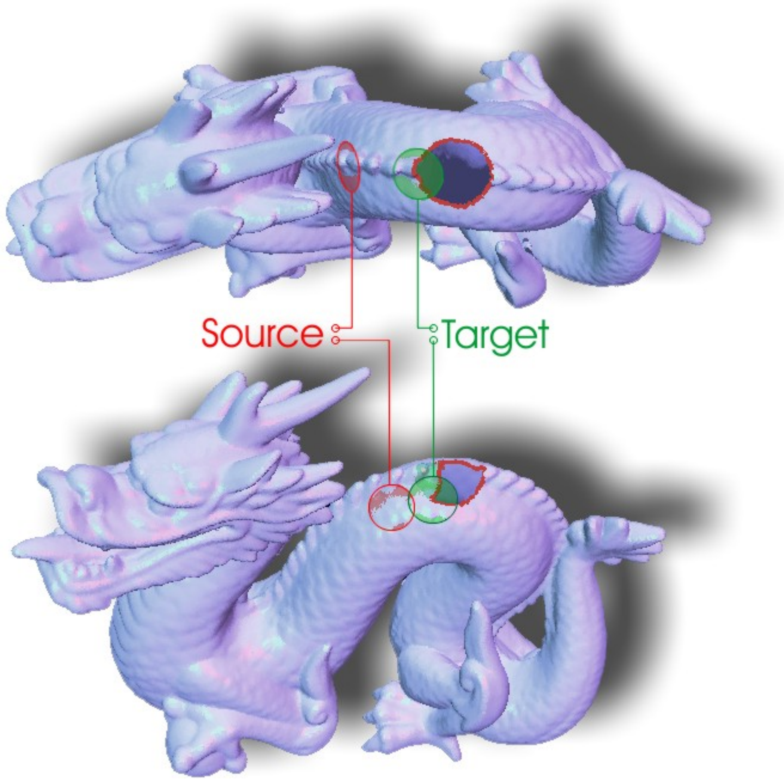


Figure 7.3: *Two snapshots from the automatic completion phase. In this phase, matching source fragments are identified and copied to the defective surface region.*

with insufficient sampling, and then by copying appropriate local neighbourhood patches (so-called *Fragments* or *Feature Surface Elements* as introduced in chapter 4.2) from a candidate set to the hole region. This way the hole is iteratively closed. As newly inserted points have an influence also on later filling steps, every point in the point set is assigned a confidence value, which is equal to 1 for all points in the original point set and is in the interval $[0, 1)$ for inserted points. By these means, it is possible to evaluate each point's confidence and adjust its influence on the algorithm appropriately.

In accordance with the terminology in image completion approaches, we call the regions close to insufficient sampling *Target Fragments* and regions from where points to be inserted are drawn

```
Fill (Point Set Hierarchy $\mathcal{P}^L, \ldots, \mathcal{P}^0$)
compute initial guidance surface $\mathcal{G}^L$ [optional]
for all $l = L, \ldots, 0$ do
    $\mathcal{B}^l \leftarrow$ find boundary points in $\mathcal{P}^l$
    $\mathcal{C}^l \leftarrow$ find candidate points in $\mathcal{P}^l$
    compute descriptors $\chi(\mathcal{C}^l)$ and $\chi(\mathcal{B}^l)$ using $\mathcal{G}^l$
    $\mathcal{Q} \leftarrow$ prioritise $\mathcal{B}^l$
    while $\mathcal{Q}$ not empty do
        $b \leftarrow \mathrm{top}(\mathcal{Q})$
        find best matching candidate $c \in \mathcal{C}^l$
        copy $\mathcal{N}(c)$ to $\mathcal{N}(b)$
        update $\mathcal{B}^l$ and $\mathcal{Q}$
    end while
    $\mathcal{G}^{l-1} \leftarrow \mathrm{MLS}(\mathcal{P}^l)$
end for
```

are called *Source Fragments* or *Candidate Fragments*. In order to assess a fragment's appropriateness for a specific filling operation and to select the best fragment from the candidate set, target and candidate descriptors are constructed as regularly sampled height fields. So-called *Guidance Surfaces* are used to derive descriptor values in invalid target regions.

With the notation given in table 7.1, the basic workflow of our algorithm can best be seen in pseudo-code (see above). The overall approach is hierarchical, i.e. it reconstructs the surface in the hole region on coarse scales first and exploits the result to derive the guidance function for the next levels. Hence, the first step in our algorithm is to compute a point set hierarchy, consisting of L point sets $\mathcal{P}^0, \ldots, \mathcal{P}^L$, where $\mathcal{P}^0$ is the original point set and $\mathcal{P}^1, \ldots, \mathcal{P}^L$ are smoothed and (optionally) subsampled copies thereof. This process is illustrated in figure 7.12 further back in this chapter. For clarity of presentation, though, a non-hierarchical, 1-level-formulation of this approach is described first, before the hierarchical formulation is motivated and presented in section 7.5.

7.4 Non-Hierarchical Formulation

Let $\mathcal{P}$, as usual, denote the given, possibly incomplete, discrete point sampling $\{\mathbf{p}_1, \ldots, \mathbf{p}_N\} \subset \mathbb{R}^3$ of an unknown surface. Following the notion from 2D-image synthesis, every point $\mathbf{p} \in \mathcal{P}$ defines in conjunction with a local frame F_p and a radius $\varrho \in \mathbb{R}$ a corresponding *surface fragment* $\mathcal{N}_\varrho(\mathbf{p}) \subset \mathcal{P}$ as

$$\mathcal{N}_\varrho(\mathbf{p}) = \{ \, \mathbf{p}_i \in \mathcal{P} \mid d(\mathbf{p}, \mathbf{q}(\mathbf{p}_i)) \leq \varrho \, \} \,, \tag{7.1}$$

where $\mathbf{q}(\mathbf{p}_i)$ is the projection of $\mathbf{p}_i$ into the plane defined by F_p (see figure 7.4). The defining local frame is established using the best fitting plane to the k nearest neighbours of $\mathbf{p}$, as suggested in [Hoppe et al. 1992]. This plane is also used as parameter plane for the fragment and its normal as surface normal in $\mathbf{p}$. Given this frame, the points in the fragment can be efficiently collected traversing a proximity graph as proposed in [Klein & Zachmann 2004].

Please note that the terms *fragment* and *local neighbourhood* are used synonymously throughout this chapter, and that the index ϱ is suppressed in unambiguous cases, as is the index l.

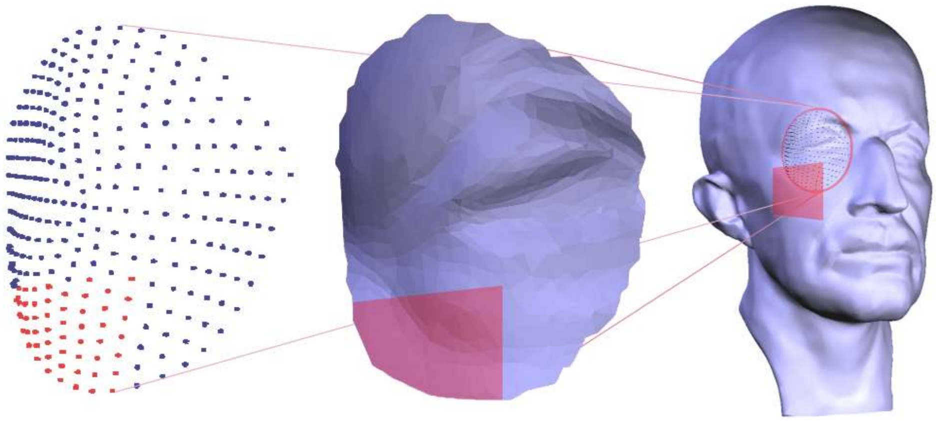


Figure 7.4: *Illustration of a local point set neighbourhood (triangulated for display purposes, centre) and its regularly resampled counterpart (left). Hole regions in the original surface (red square) lead to invalid descriptor components (coloured in red).*

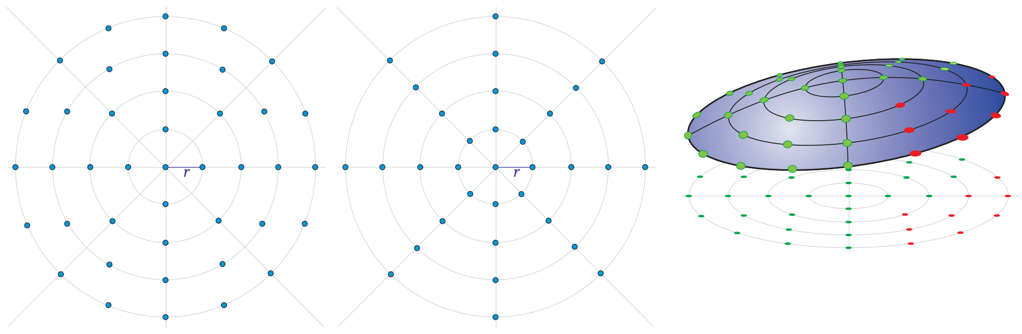


Figure 7.5: *Descriptor layout.* Left: *The number of sample points per ring grows linear with respect to its length, i.e. the sampling rate for each ring is identical (Four samples per $2\pi r$ in the depicted case).* Centre: *Descriptor as suggested in [Zelinka & Garland 2004]; here, the number of sample points per ring is constant, such that the sampling rate decreases linearly.* Right: *3D illustration of the local resampling.*

7.4.1 Neighbourhood Descriptors

Unlike 2D image fragments, 3D surface fragments as defined in eqn. (7.1) constitute an irregular and unstructured sampling of a surface. As a consequence, there is no canonic distance measure to quantify the alikeness of two fragments. Therefore, a neighbourhood descriptor (together with a corresponding distance function) has to be defined. In a recent approach, Zelinka et al. [Zelinka & Garland 2004] have shown so-called *Geodesic Fans* to faithfully identify regions on a surface that are geometrically similar. Their descriptor is a vector of D discrete samples of one or several signals defined on the surface. The samples are taken at D fixed sample positions according to some sampling pattern given in geodesic polar coordinates (figure 7.5, middle).

Dealing with point sets, the computation of geodesics is an ill-defined and expensive operation. Nevertheless, the geodesic fan approach can be adopted to the present setting by deriving a locally regular resampling of the point cloud that can then be used to come up with a straight-forward descriptor. To this end, the surface fragment is resampled according to the sampling pattern depicted in figure 7.5 (left) since it does not emphasise the regions close to the centre as the traditional geodesic fan approach (middle) does. The D sample positions can be understood as polar coordinates in the parameter plane described above, scaled to fit into the parameter domain of the fragment. A *fragment shape descriptor* is now easily defined as the vector of height values of this local regular resampling over the parameter plane:

$$\chi(\mathbf{p}) = (h_1^{\mathbf{p}}, \ldots, h_D^{\mathbf{p}})^t \in \mathbb{R}^D. \tag{7.2}$$

Please note that although on coarser scales where fragments are of larger size (see below) a considerable number of points may have to be resampled, the descriptors can be efficiently computed using natural neighbour interpolation techniques that are performed directly on the GPU [Hoff et al. 1999]. Alternatively, interpolated height values for the new sample points can also be computed quite efficiently by constructing a 2D Delaunay triangulation and using the barycentric coordinates of the sample positions for interpolation.

An obvious choice for a distance metric on the space of descriptors would be the weighted *Mean Squared Error*

$$\delta(\chi(\mathbf{p}), \chi(\mathbf{q})) = \sum_i w_i \|h_i^{\mathbf{p}} - h_i^{\mathbf{q}}\|^2, \tag{7.3}$$

with some appropriate weights w_i. However, as the sampling pattern is uniquely defined only with respect to the normal direction, position and size, leaving rotations about the normal as an additional degree of freedom. Before evaluation of equation (7.3), one has hence to allow for a set of transformations φ corresponding to discrete rotations and mirroring to be applied to the descriptor, which is henceforth denoted as χ_φ. After rotation, however, the sampling point positions of $\chi(\mathbf{p})$ and $\chi_\varphi(\mathbf{q})$ will generally not coincide; height values of at least some of the sampling points have to be interpolated. For the sake of simplicity and because it better reflects the circular structure of this descriptor, this interpolation, where necessary, is performed linearly along the rings.

Obviously, for some points in the data set, the descriptor domain will stretch into regions containing points with confidence value < 1 (points that have been inpainted in some previous step)

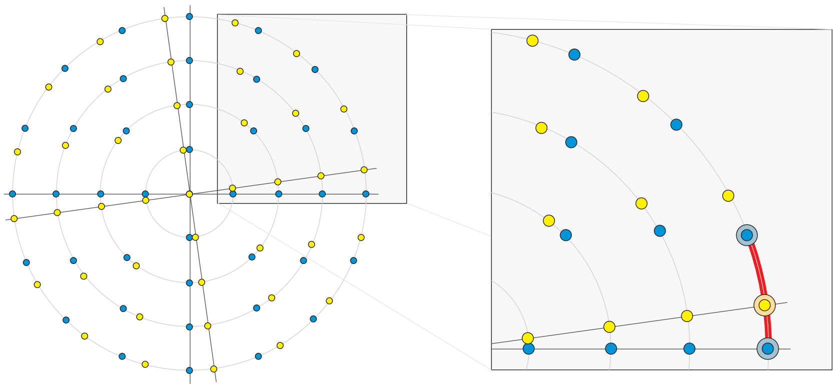


Figure 7.6: *Interpolating descriptor values. When comparing rotated descriptors (target in yellow, candidate in blue), descriptor values are interpolated linearly along the rings (highlighted in red in the zoom-out).*

or no points at all. The basic idea is that descriptor values corresponding to these regions should have less influence on the process of selecting of appropriate target-candidate pairs. This is achieved by setting the weights in equation (7.3) to

$$w_i = \alpha_i(\mathbf{p}) \cdot \alpha_i(\mathbf{q}), \tag{7.4}$$

where the $\alpha_i(\mathbf{p})$ are the interpolated confidence values of the points in $\mathcal{N}(\mathbf{p})$. Descriptor values corresponding to empty regions in $\mathcal{N}(\mathbf{p})$ are assigned a confidence value of zero.

Summarising, the similarity criterion can be formulated as follows

$$\delta(\chi(\mathbf{p}), \chi(\mathbf{q})) = \min_{\varphi} \bar{\delta}(\chi(\mathbf{p}), \chi_{\varphi}(\mathbf{q})), \tag{7.5}$$

with

$$\bar{\delta}(\chi(\mathbf{p}), \chi(\mathbf{q})) = \frac{1}{\sum_j \alpha_j(\mathbf{p})\alpha_j(\mathbf{q})} \sum_{i=1}^{N} \alpha_i(\mathbf{p})\alpha_i(\mathbf{q})(h_i^{\mathbf{p}} - h_i^{\mathbf{q}})^2.$$

7.4.2 1-Level Inpainting

Before the missing surface in hole regions can be reconstructed, the hole boundaries have to be detected using the method described in chapter 6. The basic concept is now to find for every boundary point $\mathbf{b} \in \mathcal{B}$ an appropriate candidate $\mathbf{c} \in \mathcal{P}$ and to copy its neighbourhood to the invalid (empty) parts of $\mathcal{N}(\mathbf{b})$. To guarantee that invalid regions in $\mathcal{N}(\mathbf{b})$ can indeed be filled with the corresponding regions in $\mathcal{N}(\mathbf{c})$, the candidate set $\mathcal{C}$ is built by collecting all points $\mathbf{p} \in \mathcal{P}$, whose descriptors do not contain any invalid components, i.e.

$$\mathcal{C} = \{ \, \mathbf{c} \in \mathcal{P} \mid h_i^{\mathbf{c}} \text{ valid } \forall i \, \} \,. \tag{7.6}$$

With a suitable candidate set and a discriminative descriptor at hand, the first and maybe most important part of inpainting – the detection of similarities and symmetries – quite simply consists of finding the best matching candidate

$$\mathbf{c}_b = \min_{\mathbf{c} \in \mathcal{C}} \quad \delta(\chi(\mathbf{b}), \chi(\mathbf{c}))$$

for any boundary point $\mathbf{b} \in \mathcal{B}$ and co-aligning $\mathbf{c}_b$'s local frame with the frame of $\mathcal{N}(\mathbf{b})$.

For alignment, the surface normals are mapped onto each other and the minimising transformation from equation (7.5) is applied. In order to compensate for little deviations that might result from the discreteness of the set of allowable transformations φ, an additional ICP step can be applied for the best matching candidate. Taking into account the descriptor samples only and using fixed correspondences, this can be performed very efficiently.

Finally, all points from $\mathcal{N}(\mathbf{c})$ corresponding to invalid regions in $\chi(\mathbf{b})$ are inserted into $\mathcal{P}$, receiving an aggregated confidence from equation (7.7) (see below) of the target descriptor used to compute the match. Afterwards the sets $\mathcal{B}$ and $\mathcal{C}$ are updated.

In order to reduce the time required for searching a best matching candidate, the tree structured vector quantisation method (TSVQ, [Wei & Levoy 2000]) is applied to the candidate set. By means of the TSVQ the search for a best matching candidate is significantly accelerated and renders the filling procedure interactive even for large candidate sets. For very large data sets, the performance can be further increased by early rejecting candidate fragments with inappropriate size. Thus, the minimising transformation needs only be computed for compatible candidate frag-

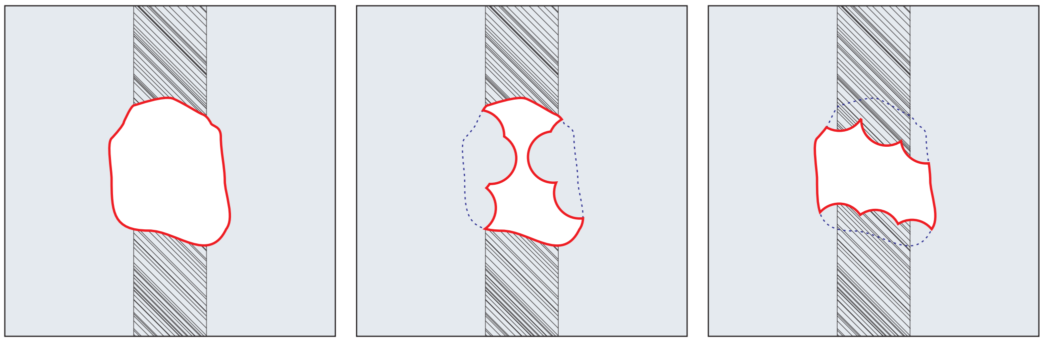


Figure 7.7: *Completing damaged images* (left) *at target points in random order may lead to undue propagation of flat areas* (middle)*, whereas only a different completion order may already lead to drastically better results* (right)*.*

ments. This will be further discussed in the hierarchical setting described below.

7.4.3 Structure-Driven Inpainting

The algorithm described so far chooses target fragments to be filled in random order. Analogously to what was noted for images by Criminisi et al. [2003], this often has the adverse effect that flat surface regions are unduely propagated into the hole region (see figure 7.7). Our algorithm tackles this by assigning priority values to all possible *targets* and performing a best-first filling algorithm. The priority values are computed to favour those targets which are on the continuation of strong features and are surrounded by a high confidence neighbourhood.

Unfortunately, feature detection on point sets in itself is a non-trivial task, let alone in the presence of holes. Instead, a simple heuristic to measure the expressiveness of a fragment can be used, that basically consists of computing the standard deviation σ of the descriptor values along the sampling rings depicted in figure 7.6. By means of this criterion, regions of high curvature are preferred over flat regions. Please note that this criterion is well-adapted to our hierarchical setting described in section 7.5, as here the fragment size corresponds to the amount of detail contained

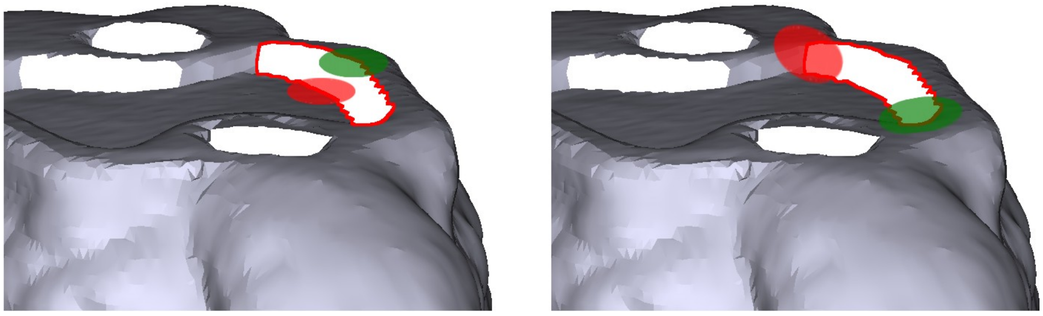


Figure 7.8: *Surface completion analogous to figure 7.7. In random order, flat regions might unduely be prolonged* (left)*, a problem that can be tackled by selecting more expressive locations first for filling* (right)*.*

in the fragment and therefore σ gives a good indication of the probability of the presence of a feature on the respective scale.

As pointed out already in [Drori et al. 2003], it is sensible to compute the confidence of insertion operations and to keep track of the aggregated confidence of inserted points. To this end, the confidence is computed once for the complete (best matching) candidate fragment:

$$\alpha(\chi_{\mathbf{p}}) = \frac{1}{D} \sum_i \alpha(h_i^{\mathbf{p}}). \tag{7.7}$$

This confidence value is assigned to each inserted point.

Experiments with several combinations of the two criteria α and σ to prioritise the filling operations have shown that a threshold approach performs best: Among those target descriptors that have the highest confidence value (quantised into ten bins), the one with highest σ is selected to be filled first. This way, of those target fragments with a high confidence we favour the most discriminative. As a consequence, problems as illustrated in figure 7.8 can be solved. In this case the algorithm correctly chooses the target fragment indicated as a green disc in the right image to be processed first, although it has about the same confidence as the green disc in the left image.

It is worth noting that the presented algorithm can of course trivially be modified to a semi-automatic approach, where a few appropriate candidates are suggested to the user, who then selects the one to be pasted into the target region.

7.5 Hierarchical Formulation

The essence of exemplar-based completion is to exploit coherence and similarity between the region of interest and appropriate candidate regions of the considered object. Geometric properties of the hole region, though, might be represented in different scales, and in many cases similarity relations present in different scales correspond to very different regions on the object. It is therefore important to allow candidates to stem from the optimal object region *per scale*, such that for instance the coarse geometry of the bunny's missing left knee (see figure 7.12) is reconstructed on coarse levels by copying the corresponding geometry from the bunny's right knee, whereas the fur structure, exhibiting different similarity relations, is reconstructed from various different locations on the bunny (including the bunny's right knee but also from the back and front).

Only in the presence of real symmetry, where similarity on all considered scales happens to relate to the same candidate region, the one-level approach described in the previous section is generally sufficient. For instance, the missing left eye and nose region of the Max-Planck-model (as shown in figure 7.13) can be reconstructed using transferred and mirrored copies of parts of the opposite side of the face. This type of similarity relation ranging over all considered scales – rarely encountered when dealing with images – is relevant for large classes of 3D objects. Nevertheless, in order to handle cases as described above, we propose a hierarchical, multi-level approach, whose first step is to create a point set hierarchy $\mathcal{P}^1, \ldots, \mathcal{P}^L$ with according scales $\varrho_1, \ldots, \varrho_L$, each point set representing the (defective) object and its geometrical properties up to the according scale.

7.5.1 Creating the Point Set Hierarchy

The scale-space representation of the input model required to be able to separate features with respect to their scale is constructed by iteratively applying Laplacian diffusion, deriving smoother and smoother versions of the object under consideration, corresponding to ever larger kernel widths. Specifically, to derive $\mathcal{P}^{l+1}$ from $\mathcal{P}^{l}$ we compute new point positions as the weighted mean of all k nearest neighbours $\mathbf{p}_j^l$ of $\mathbf{p}^l$:

$$\mathbf{p}^{l+1} = \frac{1}{\mu}\sum_{j=1}^{k} \mu_j \mathbf{p}_j^l, \tag{7.8}$$

where $\mu = \sum_j \mu_j$. (Actually, we perform the smoothing in direction of the surface normal only, as we wish to smooth the surface itself, rather than the distribution of sample points *in* the surface). This corresponds to smoothing $\mathcal{P}^l$ with a kernel width of

$$\varrho_{\mathbf{p}} = \max_{j=1,\dots,k} d(\mathbf{p}, \mathbf{p}_j^l).$$

The average distance to the k^{th}-nearest neighbour

$$\varrho = \frac{1}{N}\sum \varrho_{\mathbf{p}}$$

is called the *k-Ring Radius* and describes a natural size of the local neighbourhood patches, as it contains all the detail information up to the respective hierarchy level, with all higher level detail information smoothed out (see figure 7.9).

On first sight it appears that the smoothing scheme described above has two main drawbacks: On the one hand, it is a well-known fact that Gaussian filtering causes shrinkage (that ultimately would reduce the data to a single point). However this is of no negative effect for the present purpose, as all descriptor comparisons are evaluated separately within one scale and consequently any potential shrinkage is cancelled out. On the other hand, the smoothing is not invariant with respect to the sampling density since the points contributing to the new, filtered point positions are a fixed number of nearest neighbours. Strictly speaking,

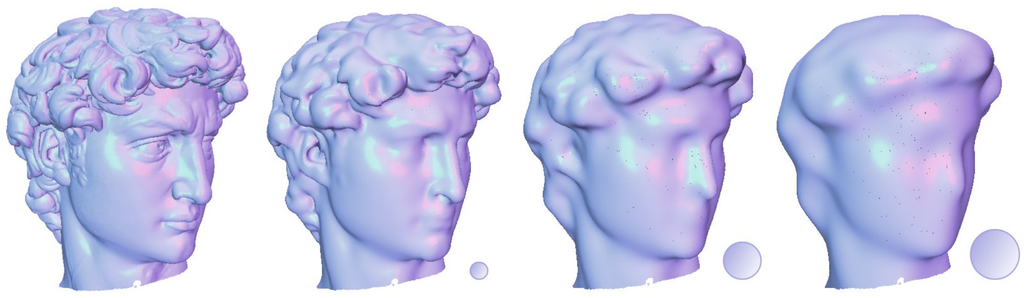


Figure 7.9: *David's head, subsampled to 300000 points, original* (left), *iteratively smoothed once with $k = 100$* (mid-left), *after 5* (mid-right), *and 8 iterations* (right). *The discs indicate the corresponding neighbourhood size.*

claiming that a certain "scale" is represented in a smoothed point set, therefore holds only for at least roughly uniformly sampled point sets. To address this drawback, more sophisticated filtering methods could be applied, in the spirit of [Karbacher & Häusler 1998; Desbrun et al. 1999; Clarenz et al. 2000] or [Taubin 1995]. In practice, where objects to be repaired stem from laser range scanners or comparable acquisition techniques and are thus very densely sampled, a uniform subsampling may be applied for scale space generation. Please bear in mind, that the smoothed versions constitute temporary intermediate representations of the object and are used for guidance only.

7.5.2 Multi-Level Inpainting

Based on the point set hierarchy $\mathcal{P}^0, \ldots, \mathcal{P}^L$, the inpainting is a bottom-up process, filling the hole in its coarsest scale representation $\mathcal{P}^L$ first and then consecutively on the finer levels up to the finest level $\mathcal{P}^0$. In each step (aside from the first step, where $\mathcal{P}^L$ is completed using the non-hierarchical formulation of our algorithm as described in section 7.4), the previous, next coarser level point set is used to construct a guidance surface that can be exploited in the target descriptors for the filling step on the current level. This way it is possible to encode hints to the larger scale geome-

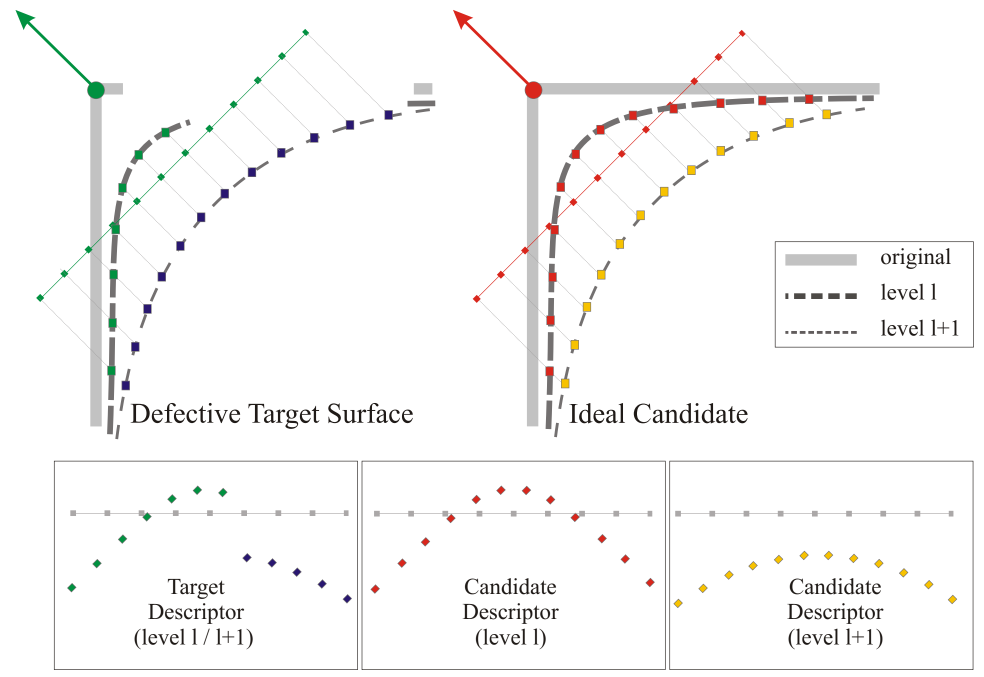

Defective Target Surface
Ideal Candidate
original
level l
level l+1
Target Descriptor (level l / l+1)
Candidate Descriptor (level l)
Candidate Descriptor (level l+1)

Figure 7.10: *Defective target surface and an ideal candidate (bold), together with two levels from the scale-space representation (dashed, level $l+1$ filled, level l incomplete). Updating target descriptor values invalid on level l using the guidance surface from level $l+1$ leads to a descriptor (bottom left) that is not well comparable with either of the candidate descriptors (bottom centre / right).*

try into the descriptor components that have been invalid till now and hence neglected.

Let the *Guidance Surface* $\mathcal{G}^l$ be any implicit representation of the (completed) point set $\mathcal{P}^{l+1}$. In this approach, the zero set of the MLS-approximation of $\mathcal{P}^{l+1}$ is used, but any other locally evaluable representation could also be applied. A straight-forward approach, that would also resemble comparable approaches in 2D image processing, would then be to assign height values to invalid target descriptor components by sampling $\mathcal{G}^l$ (see figure 7.10, bottom left). This straight-forward approach, however, would have the adverse effect that even ideal candidates would not be considered a perfect match (figure 7.10, bottom centre). The reason for this is that inserting samples from $\mathcal{G}^l$ to the current level's descriptor inherently causes two scales to be mingled. The resulting *hybrid* descriptor – incorporating two scales at the same time – is in fact comparable to descriptors on *neither* the current level l *nor* the coarser level $l+1$ (figure 7.10, bottom right).

7.5.3 2-Layer Descriptor

As a consequence, another type of descriptor is required that is able to differentiate between the respective scales. Consider the 2-layer descriptor illustrated in figure 7.11:

- The first layer χ_l is constructed as described in section 7.4.1, capturing the available local geometry from $\mathcal{P}^l$ only, and assigning zero confidence to the invalid descriptor components.
- For the second layer χ_{l+1}, the same parameter plane and the same sampling pattern are used, but height values are derived from the zero level set of the MLS-approximation of $\mathcal{P}^{l+1}$.

The distance function for the two-layer descriptor is then simply a weighted sum of the level-wise distance functions:

$$\delta(\chi(\mathbf{p}), \chi(\mathbf{q})) = \delta(\chi_l(\mathbf{p}), \chi_l(\mathbf{q})) + \tau\, \delta(\chi_{l+1}(\mathbf{p}), \chi_{l+1}(\mathbf{q})). \tag{7.9}$$

While the parameter τ is arbitrary in principle, a value of 0.3 has proven to produce good results in all experiments performed. In cases where multiple hierarchy levels are reconstructed, it is advisable to increase τ for finer levels, as they can be expected to be already a reliable reconstruction.

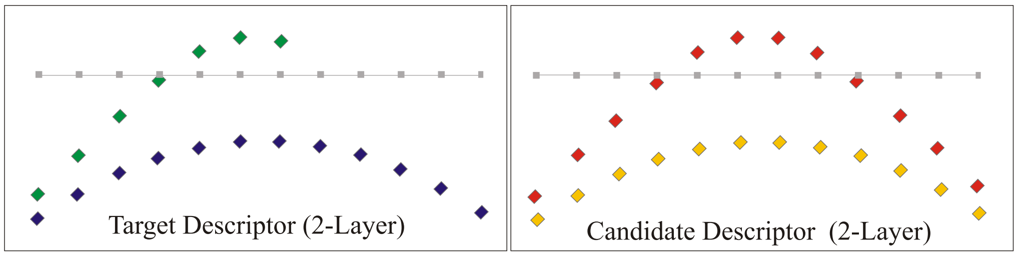


Figure 7.11: *The 2-Layer descriptor for the situation in figure 7.10*

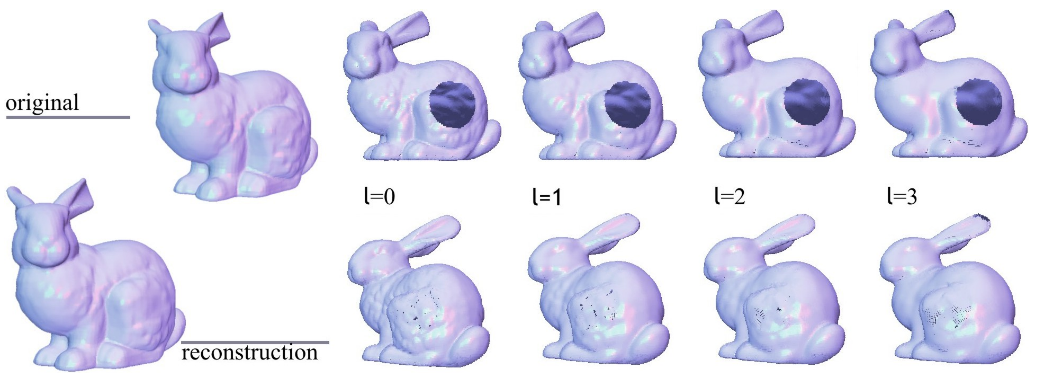


Figure 7.12: *Hierarchical reconstruction of the Stanford Bunny. First, a point set hierarchy ($l = 0$ to 3) of the defective bunny is constructed (top). Starting with $l = 3$, each level is filled per-se, where the level $l + 1$ serves as Guidance Surface for the filling of level l.*

As stated above, the coarsest level L is filled without guidance (formally setting $\tau = 0$), as there is no previous reconstruction available for evaluation as guidance surface. Given that the coarsest level's scale size corresponds well to the scale of the hole, this problem is sufficiently well-posed. However, for very large holes, considerable filtering might be necessary to this end. In this case a natural and trivial extension to the presented algorithm is to use any one of the available smooth hole filling schemes for the coarsest level and use the result as guidance surface $\mathcal{G}^L$.

7.6 Results and Conclusions

We applied our fragment-based inpainting algorithm to various data sets of point sampled geometry. The objects depicted in the images of this chapter exhibit holes in structured surface regions and are in addition to this comparably large in size. Reconstructing the surface for these holes using traditional smooth hole filling algorithms would have lead to disturbing visual artefacts.

Figure 7.13 illustrates the basic workflow of the presented algorithm. For target fragments (illustrated as green discs) an optimal candidate fragment (red discs) is identified. The points corresponding to invalid target regions are pasted into the point set af-

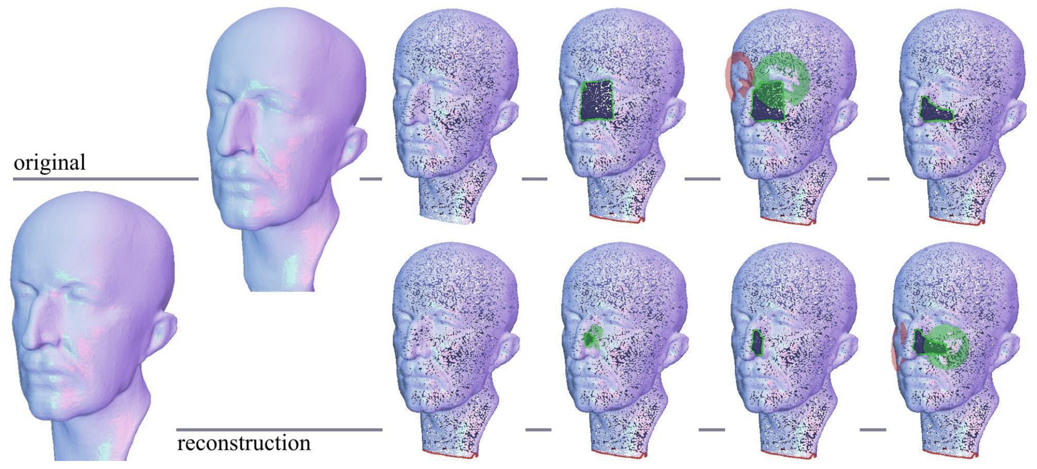


Figure 7.13: *Original Max Planck data set* (top left)*, and the successive steps towards the final completion* (bottom left)*. The large images to the left are rendered from reconstructed triangle meshes from the original and the completed point sets, respectively.*

ter the according transformations (translation, rotation, optional mirroring), which are deduced from the descriptor comparisons, are applied. In near symmetric cases like human faces, the non-hierarchical formulation already gives satisfying results, given that the required scale to cover the hole can be represented without scale-space segmentation. Figure 7.14 gives another example of near symmetry that is often encountered in particular for models of human faces and can therefore be exploited for completion in a non-hierarchical fashion.

Figure 7.12 demonstrates the use of the hierarchical formulation for the exploitation of similarities that are spread over several scales. The presented approach is able to reconstruct both the knee as a symmetrical large scale feature and the fur structure that itself does not exhibit an analogue symmetry, but is also well presented as a coherent feature on the surface on finer scales. During the coarse level inpainting steps, corresponding target-candidate descriptor pairs were identified. In this example, prioritising the target fragments for filling according to their discriminativity was particularly useful. This way, the target regions close

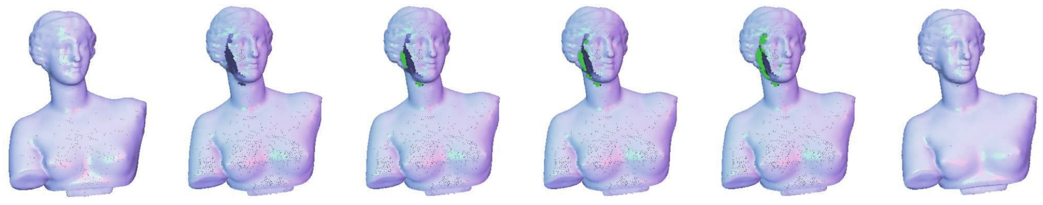


Figure 7.14: *Original Aphrodite data set* (left), *an incomplete scanning session and its iterative completion. Inserted points are visualised in bright green, flat shading.*

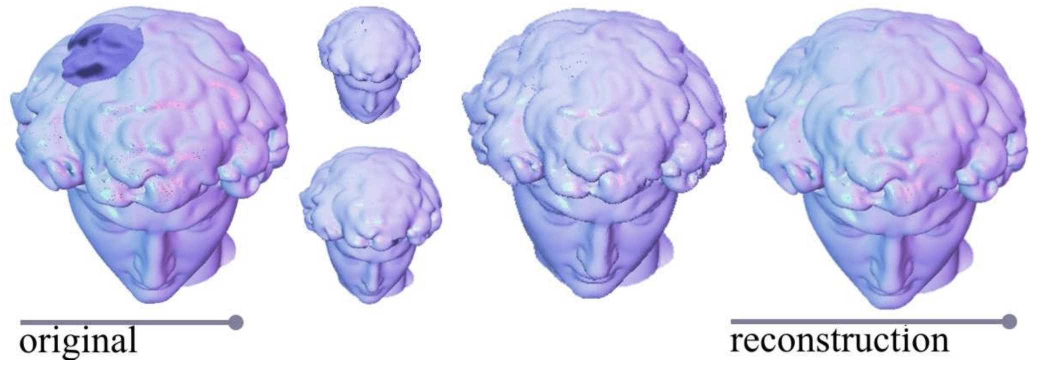


Figure 7.15: *Hierarchical reconstruction of the David Head Model.*

to the bunny's knee were selected for filling first. During the finer scale filling operations, the fur structure was transferred to the hole region from various (other) regions on the bunny's back.

Also, the david head model from the Michelangelo project (figure 7.15) could not have been filled using a 1-level approach, as the model itself does not contain appropriate candidates that correspond to the (unknown) hole regions' full spectrum of scales. By filling the hole for coarser regions, representing the basic geometry, first, and adding more and more detail with decreasing neighbourhood size at later stages, our algorithm was able to inpaint this hole in a visual believable way, taking into account the objects global and local context. Nevertheless, this example also demonstrates the limits of the algorithm as presented in this chapter. Features for which no similarity relation exists in the object itself cannot be reproduced, such that one of david's more promi-

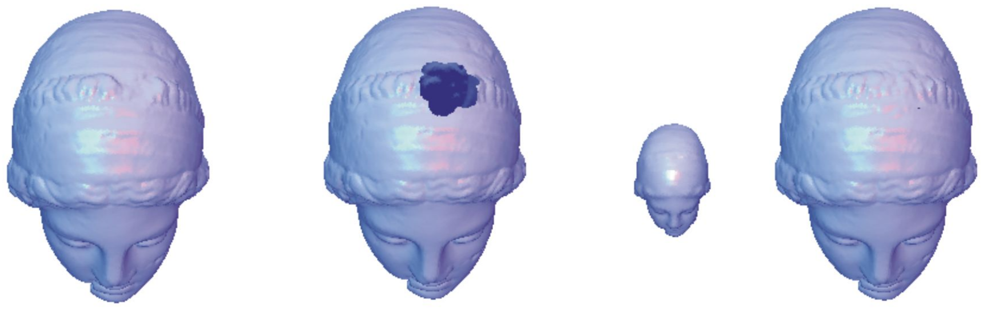


Figure 7.16: *Igea data set.* From left to right: *Original data set, reduced model, completed model on coarse level (no guidance surface was available for this completion), and completed model on fine scale. Please note that for the coarsest level, the point set was decimated and the image is scaled here correspondingly for visualisation purposes. Only the point sets themselves were rendered (no splatting).*

nent curls is missing in the inpainted region. This topic will be discussed further in chapter 9.

Further successful examples of the hierarchical completion scheme are depicted in figures 7.16 and 7.18.

In order to assess the influence of the automatically computed guidance surface and the candidate set on the inpainting results, we reconstructed the bunny data set with the help of the complete point set itself as guidance surface and candidate set. The combination of both, perfect guidance and perfect candidate set, resulted in the perfect reconstruction of the bunny (figure 7.19). As opposed to that, figure 7.12 shows the hierarchical reconstruction of the incomplete bunny without any additional knowledge.

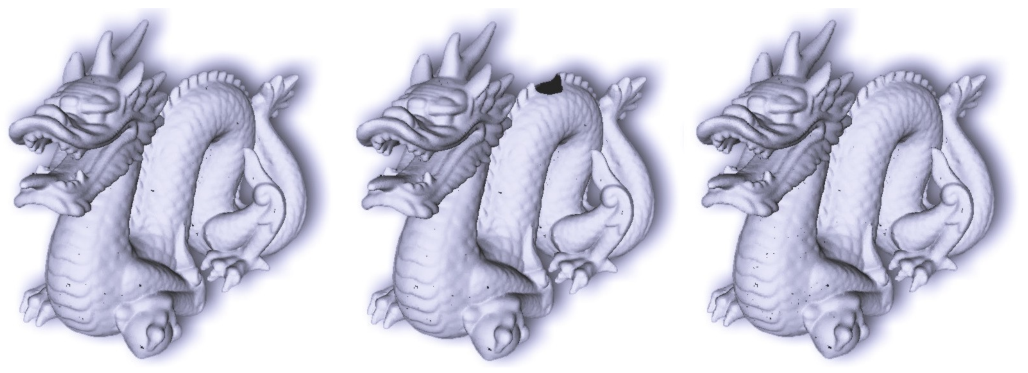


Figure 7.17: *Dragon data set.* From left to right: *Original data set, reduced model, completed model.*

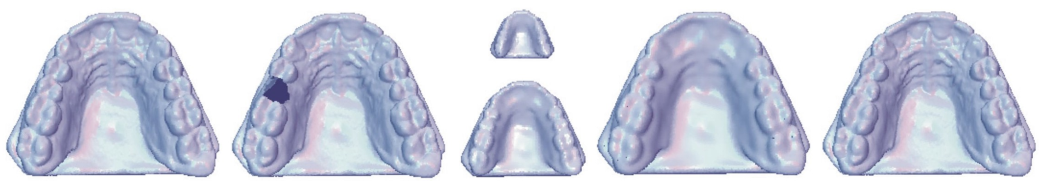


Figure 7.18: *Teeth data set.* From left to right: *Original data set, reduced model, iterative repair steps, resulting reconstruction.*

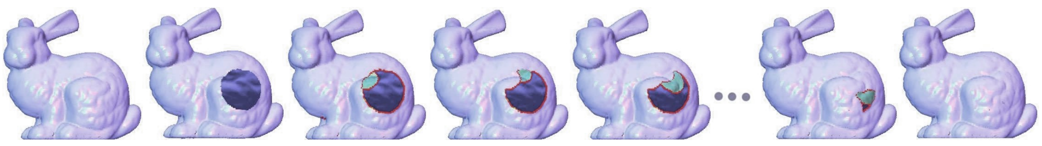


Figure 7.19: From left to right: *Original Bunny data set, data set with artificially introduced hole, iterative repair steps, resulting reconstruction. In this example, the candidate set and the guidance surface have been built from the original bunny data set.*

Part IV

Modelling

Chapter 8

3D Shape Modelling

The preceding parts of this thesis are concerned with, one could say, the mere reconstruction of existing geometry in a manner as accurate and reliable as possible, opening the paths to numerous applications like documentation, presentation, archiving, analysis, and in cases even physical reproduction. However, the spectrum of options available with digital 3D objects that are not encumbered with the restrictions inevitably tied to their real counterparts is not exhausted by these applications.

One of the options neglected so far in this thesis is creative design. At first sight the concept of creative design might seem *incompatible* with the needs and requirements typically linked to cultural heritage applications. But there are indeed numerous situations where it is mandatory to set aside the rule of ”sticking to the truth” and let a user's creativity, knowledge, and understanding of a scene or an object influence and modify the digital object.

Museum curators and art professionals do not only conserve the artefacts under their auspices but also perform restoration to exhibit the most precious pieces of art in their former glory. It is not only analogously, but indeed to a much greater extent, that digital objects should be available for easy repair, modification, or transform, given that the digital original can always exist in multiple instances and hence be retained *and* modified – a fact that makes them distinctively more versatile compared to their real counterparts.

8.1 Related Work

Traditionally, the goal of research in geometric modelling and design was to find representations of 3D objects and modelling tools enabling a possibly intuitive and efficient design process, the choice of the former usually influencing the range of choices for the latter and vice versa. In other words, the way an object is defined naturally and inevitably influences the way it can be handled.

Originally, the needs of the industry to manufacture products with well-defined physical properties imposed requirements on the geometric design methodology which made modelling a cumbersome and time consuming process. A prominent example of *free form surface design* methodologies that developed fairly early in particular in the automotive industries are the piecewise polynomial tensor product surfaces (see [Farin 1990] for an early overview). Here, even the changes appearing conceptually small are often hard to execute and require a considerable amount of expertise. However, exact and provable geometric properties of the product are not relevant in numerous current application areas like the film industry, computer games, etc. Since the most relevant quality measure in these contexts is the visual appeal, much of the research focusses on finding new modelling metaphors and techniques that allow designers to work efficiently, which calls for intuitiveness and simplicity, especially in the early stages of design.

The aforementioned dependence of the modelling interaction on the specific object representation is one of the key hindrances in intuitive editing, as animadverted already by [Welch & Witkin 1992]. Optimally, the specific object representation should be invisible to the user and not apparently limit the range of possible modelling operations (as is the case with the aforementioned tensor product surfaces whose degree defines the number of control points to be positioned in order to derive the desired modification and vice versa.)

Among the earliest modelling approaches that try to sever the ties between object representation and the modelling methodology are the so-called implicit deformation schemes. In these, the objects are embedded in a space which is warped by dragging a sparse set of control points (e.g. the corners of an embedding cube). The warping of the embedding space causes a corresponding deformation of the object. According to the most popular such approach called Free Form Deformations (FFD) [Barr 1984; Sederberg & Parry 1986], the surrounding space is defined as a multidimensional spline. This technique has been further improved and generalised [Coquillart 1990], moreover, it was adapted to generate animations by e.g. [Coquillart 1990; Faloutsos et al. 1997]. The main advantage of these methods is that they enable a deformation that is independent of the complexity of the object being manipulated. However, as pointed out in [Hsu et al. 1992], the placement and control of the lattice defining the deformation is non-trivial. The improvements of [Hsu et al. 1992; Hu et al. 2001] remedied these problems for the case of a single edit. In this case the user has to define only the initial control lattice, which is then modified invisibly to the user as he edits the object by dragging a point on the surface. However, since the control points move according to the user's input, the result of a subsequent edit depends on a new control lattice, and may thus be contra-intuitive, as it is different from an identical edit in an untouched region, cf. [Frisch & Ertl 2002].

The algorithmic generalisation of piecewise polynomial representations are the subdivision surfaces [Chaikin 1974; Catmull & Clark 1978], where a smooth surface is generated by iterative refinement of a control mesh according to certain subdivision rules. Since subdivision surfaces essentially comprise a representation of an object on different levels of scale, they may be utilised as a basis for an editing concept called Multi-resolution Mesh Editing (MME) [Zorin et al. 1997]. The idea of multi-resolution editing is to use different levels of detail of the object to perform edits on different scales: Detail edits are performed on finer meshes and large scale edits on coarser versions of the mesh. Saving the finer meshes as details with respect to the coarser meshes provides for detail preservation during large scale edits. Unfortunately, since the one-ring of the edited vertex defines the region of influence (ROI) of an edit, the subject of the edit and the ROI are com-

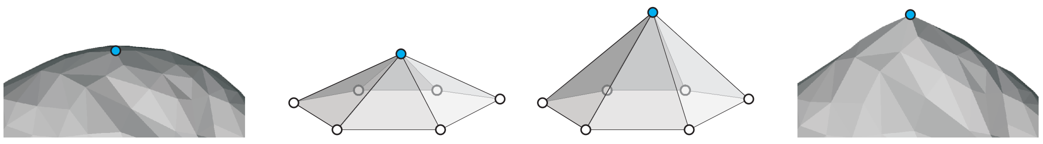


Figure 8.1: *Illustration of an elementary multi-resolution mesh edit. From left to right: Part of the mesh to be modified, a part of the corresponding coarse mesh, modified coarse mesh, modified mesh. In this type of edit, the one-ring of the transformed vertex in the coarse mesh defines the region of influence of the editing operation. The support of an editing region can thus only be selected by choosing a corresponding coarse mesh representation, if available.*

pletely connectivity-defined and cannot be chosen arbitrarily (see figure 8.1). Kobbelt et al. [1998; 2000] provided a solution to this problem by abandoning the idea of defining the multi-resolution representations as a coarse-to-fine hierarchy of nested meshes, and rather utilising a hierarchy of scales in terms of smoothness. The levels of detail are generated by a hierarchical mesh smoothing scheme.

In extension of an idea presented in [Klein 2000], Kobbelt et al. [2000] also describe an important modelling metaphor that could be considered mainstream today: During the editing process the designer picks a region of interest (which in this particular approach also defines the scale of the edit) and a handle, i.e. another region within the ROI that rigidly undergoes the user-defined transformation. This type of editing metaphor will be described in more detail in section 8.2. Each time the user transforms the handle, the mesh within the ROI transforms to produce a smooth transition between the rigidly transformed handle and the unmodified region.

Multi-resolution mesh editing can serve as a particular paradigm raising two main questions that drew attention in the major part of the more recent research activities in interactive modelling:

1. How to propagate the handle transformation into the region of influence?

2. How to deal with surface details? How are they defined, how should they be transformed (or maintained) during modelling?

To answer the first of these questions, many authors ([Kobbelt et al. 1998],[Guskov et al. 1999],[Botsch & Kobbelt 2004], and [Botsch & Kobbelt 2005], see also the references in [Botsch et al. 2006]) use some form or other of the *variational surface modelling* concept introduced by Welch and Witkin [1992] (cf. also the contemporary publication of Moreton and Séquin [1992]). The fundamental idea in these approaches is that the geometric layout of the deformed region should generate a *fair* surface. If nothing else is specified e.g. by the application or by the context of the editing operation, a fair surface is typically understood to be one that corresponds to the principle of the *simplest shape* [Sapidis 1994], which is typically formalised as a constrained energy minimisation problem where the transformed control region and the unmodified (fixed) surface region serve as constraints.

Defining the transformed surface as a minimal surface obviously would destroy any fine detail that might have existed in that surface region. Above approaches therefore perform a multi-resolution decomposition first to separate from the base geometry the (user-defined) amount of detail which is re-added to the transformed surface after editing. One drawback of this method is that by transforming the base surface, also the frames in which the details are defined are transformed. This might lead to un-intuitive deformations of the details. Recent years have therefore shown an increasing interest in the formulation of differential coordinates, where vertex positions are implicitly defined with respect to the surrounding mesh [Yu et al. 2004], [Lipman et al. 2004], [Sorkine et al. 2004], [Zayer et al. 2005], [Lipman et al. 2005]. For a detailed overview see [Sorkine 2005].

One remark with respect to the fair surface design mentioned in the previous section: Recent years have shown a trend to motivate the computational way the modified region is deformed with analogies from physical simulation (see e.g. [Botsch et al. 2006] and references therein). In these approaches, the deformed surface is typically defined as the minimal surface with respect to some energy functional, e.g. the thin plate energy. Although this line of argumentation seems doubtful for pure boundary representations that do not encapsulate any inner physical properties of any sort of the corresponding real object, these approaches are successful in so far, as the *plausibility* of the editing operation and the appeal of the resulting surface is concerned. One reason why this is so is that apparently users find those modelling metaphors intuitive and easy to use which deform digital models *expectably*, i.e. in a way that makes sense to the user and mimics his real life experience handling objects made of some soft material.

In addition to the aforementioned desired properties of a successful editing method, one further key feature is the intuitive and precise control, incorporated into above approaches in the form of constraints that can be specified in any number of ways. Many *direct editing* approaches require the user to pick and specify a transformation for parts of the object. Lee [1999] proposes a method where the user picks a set of handle vertices in the mesh and specifies modifications for these. For vertices in the editing region, which is defined by the user beforehand, the transformations of the handle vertices are interpolated using multilevel B-Splines. These are parameterised over a 2D embedding of the editing region, making this method suitable especially in flat regions. The influence of the handle vertices' transformation on neighbouring vertices is determined by the size of the coarsest control lattice used in the B-spline interpolation. Pauly et al. [2003] presented a modelling technique, transferring multi-resolution results to the case of point-sampled geometry. In their setting, shapes are modified by defining a so-called *zero-region* and a *one-region*. The one-region undergoes the full user-defined transformation (translation or rotation), whereas the zero-region remains fixed and a predefined blending function is used to create a smooth transition between the two regions. Also focussing on point-sampled geometry Zwicker et al. [2002] generalise standard 2D image editing techniques to 3D, reconstructing well-known pixel editing tools. Into the same class of editing methods also falls the approach of Llamas et al. [2003; 2005] which introduces a two-handed editing approach specifying two modelling constraints synchronously.

Special care is taken in this approach for cases in which the region of influence of the two constraints overlap.

Although precise control undoubtedly is an important property of many modelling algorithms, there are some occasions where precision cannot even be expected from the modelling input and therefore stands back behind speed and easiness of the editing operation. As a consequence, various authors have introduced modelling approaches based on sketching, where the new layout of the geometry is only hinted at using fast strokes of an input device. Examples of this class of methods are SKETCH[Zeleznik et al. 1996], SKIN [Markosian et al. 1999] and TEDDY [Igarashi et al. 1999], which were later extended in [Karpenko et al. 2002], and [Nealen et al. 2005]. Modifying shapes in these approaches is realised using a method called oversketching, i.e. drawing parts of the silhouette of the shape anew.

Although interactive display of *implicit surfaces* [Bloomenthal 1997] is still a challenge, implicit modelling has gained more and more research attention in recent years. Distance surfaces like *blobs* [Blinn 1982], *meta-balls* [B.Wyvill et al. 1986], *soft objects* [Bloomenthal & Wyvill 1990], and *convolution surfaces* [Bloomenthal & Shoemake 1991] are popular in Computer Animation since the geometric "skeleton", with respect to which they are defined, can be used as an internal structure to control the animation [Cani 1999] and even for LOD-representations [Cani & Hornus 2001; Angelidis & Cani 2002]. Several methods have been presented to tackle blending of the implicits in case of non-neighbouring skeleton elements coming close to each other, a problem that is often referred to as "unwanted blending problem", e.g. [Angelidis et al. 2002]. Furthermore, the availability of inside-outside information allows for efficient collision detection and response [Opalach & Cani-Gascuel 1997].

Of particular relevance for the modelling approach presented in this chapter is the work of Borrel et al. presented in [Borrel & Bechmann 1991; Borrel & Rappoport 1994] and later extended in [Raffin et al. 2000]. The algorithm proposed here is similar in the way the displacement of vertices in the neighbourhood of user-defined handles are computed but is extended to incorporate sharp features where desired. In addition to that, other parameterisations are examined and geodesic distance fields are used (where available) to define an object-inherent parametrisation for the shape functions, which are a key aspect for the flexibility and intuitiveness of the presented approach. This way the user is freed of the need to adjust object-independent ROI definitions as required e.g. in [Raffin et al. 2000].

Moreover, also anisotropic parameterisations are feasible through multiple, handle-independent *anchors*. In addition to a closed formulation for the editing method including shape function modulation, the novel method features editing occluders, i.e. implicitly defined 3D-objects that influence the editing operations. These editing occluders enable a novel editing paradigm resembling the forging process where an anvil is used to give an object the desired shape and thereby transfer the concept of *Precise Contact Modelling* (PCM) to the setting of triangular meshes.

8.2 Editing Process

This chapter deals with how the user's specification of the modification for a set of handle vertices is used to produce a smooth, detail-preserving and intuitive deformation of the whole surface. For a start, the method presented in [Raffin et al. 2000] will be briefly reviewed in the following sections. After a short description of the modelling metaphor employed here, the remainder of this section will focus on how this original methodology can be extended to rotations.

During this description it will turn out that the definition of suitable parameterisations of the editing region are a key to intuitive editing of non-trivial surfaces. The hasty reader is referred to section 8.3, however, where this will be discussed in detail. General transformations and variations of the handle editing metaphor will be subject of section 9.5, which introduces a modelling framework based on the one described here but custom-tailored for modelling tasks in the context of surface completion.

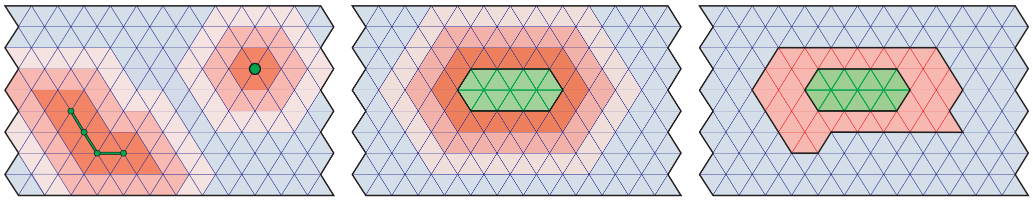


Figure 8.2: *Editing concept based on a separation of the object into a* fixed region *(blue), a* handle *(green) and the* deformed region *(red). Typical realisations of a handle are in form of single vertices, lines (both left), or surface patches (middle). Here, the region of influence of the edit is defined implicitly via distance to the handle and an additional decay function. Right: The ROI can also be selected explicitly by drawing its boundary onto the object (see section 9.5).*

8.2.1 Modelling Metaphor

Despite the apparent advantages that sketch-like editing metaphors demonstrate in particular in playing and learning environments, in this approach we stick to the method often referred to as *handle editing*. The first step of this drag and drop-like editing metaphor is to separate the object into disjoint parts: The handle, the fixed area, and the deformed area. The handle is typically defined explicitly either by selecting the respective vertices or by drawing its borders onto the surface (see figure 8.2). The user then picks and moves a number of handle vertices to specify the constraints for the transformation, while the fixed region remains unchanged.

A *plausible*[1] transition between handle and fixed region is achieved by assigning a transformation to the vertices in the deformed region that is – loosely speaking – somewhat in-between the full transformation of the handle and the identity transformation of the fixed region (see figure 8.3). What is to be understood as "somewhat in-between" will be formalised in the following sections.

[1]The term "plausible" is used here rather than the term "smooth" that is often postulated in this context, because with the editing framework described herein, the smoothness of the editing operation is completely at the user's discretion.

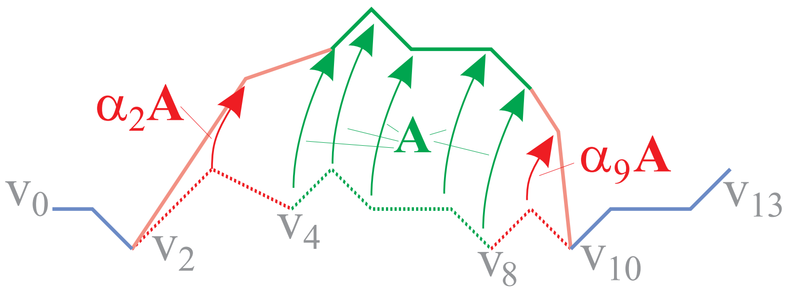


Figure 8.3: *Transformation of a simple polygonal curve. Vertices* $\mathbf{v}_4$ *to* $\mathbf{v}_8$ *form the handle and undergo the full transformation A, whereas vertices* $\mathbf{v}_0$ *to* $\mathbf{v}_2$ *and* $\mathbf{v}_{10}$ *to* $\mathbf{v}_{13}$ *remain fixed.*

8.2.2 Translations

Let us first consider the case where a user specifies a translation for the handle vertices. The main idea is that every geometric modification of a 3D shape can be interpreted as a displacement function

$$\mathbf{d} : \mathbb{R}^3 \rightarrow \mathbb{R}^3 \tag{8.1}$$

that assigns to every point $\mathbf{p} \in \mathbb{R}^3$ a displacement vector $\mathbf{d}(\mathbf{p})$ such that the resulting point positions after the modification are given as

$$\mathbf{p}_{new} = \mathbf{p}_{old} + \mathbf{d}(\mathbf{p}_{old}). \tag{8.2}$$

In our setting the values of this displacement function (also referred to as *constraints* [Borrel & Rappoport 1994]) are defined at the handle vertices only. For all other vertices the mapping has to be determined.

The idea to solve this problem is to write the displacements of all vertices as a weighted sum of virtual displacement vectors for the handle vertices – which we call *partial displacements* $\mathbf{d}_j$ in contrast to the *total displacements* $\mathbf{d}(\mathbf{p})$

$$\mathbf{d}(\mathbf{p}) = \sum_{j=1}^{k} \alpha_j(\mathbf{p})\, \mathbf{d}_j. \tag{8.3}$$

Here k is the number of handle vertices and

$$\alpha_j : \mathbb{R}^3 \rightarrow \mathbb{R}, \quad j = 1, \ldots, k$$

are weight functions. Note that for each $\mathbf{p}$, $\alpha_j(\mathbf{p})$ can be interpreted as a special weight corresponding to the handle vertex $\mathbf{p}_j$.

The above formulation allows for the desired degree of freedom for choosing the set of handle vertices and the properties of the edit. Please note that simply setting $\mathbf{d}_j = \mathbf{d}(\mathbf{p}_j)$ is not a satisfying choice, as becomes clear when we consider the case where the ROI of different handle vertices overlap in a way such that $\alpha_j(\mathbf{p}_i) \neq 0$ for a pair of handle vertices $\mathbf{p}_i$, $\mathbf{p}_j$, $i \neq j$. In this case $\mathbf{p}_i$ undergoes (in addition to its own transformation) a transformation induced by the handle vertex $\mathbf{p}_j$. This would render the handle vertices moving to positions different than defined by the user, unless we impose strict normalization conditions on the weight functions, which, in turn, would prohibit the use of arbitrary and user-defined weight functions. Therefore, we calculate the partial displacements *according to* the weight functions.

Since, by user-definition, $\mathbf{d}(\mathbf{p}_j)$ is known for the handle vertices $\mathbf{p}_1, \ldots, \mathbf{p}_k$, equation (8.3) leads to a linear system of equations

$$\Big(\mathbf{d}(\mathbf{p}_j)\Big)_{j=1,\ldots,k} = \mathbf{A}\,\Big(\mathbf{d}_j\Big)_{j=1,\ldots,k} \tag{8.4}$$

with

$$\mathbf{A} = \Big(\alpha_j(\mathbf{p}_i)\Big)_{\substack{i=1,\ldots,k\\ j=1,\ldots,k}} \tag{8.5}$$

giving us $3k$ equations for the total displacements $\mathbf{d}(\mathbf{p}_j)$ with $3k$ unknowns $\mathbf{d}_j$, $1 \leq j \leq k$.

Of course, ill-conditioned weight function choices (such as $\alpha_j(\mathbf{p}_j) = 0$) might leave the matrix $\mathbf{A}$ singular or close to singular. This can easily be avoided using either a SVD for detecting and prohibiting those ill-conditioned modifications of the shape function or ROI, or a pseudo-inverse as suggested in [Penrose 1955]. For a detailed discussion cf. [Borrel & Rappoport 1994].

After solving (8.4), (8.3) is used again to compute the total displacements for the other vertices.

Inverting the matrix $\mathbf{A}$ in equation (8.4) might seem steep in an interactive editing environment; it is therefore important to mention that $\mathbf{A}$ has dimensions $k \times k$, where k is the number of handles and does thus not depend on the overall size of the object.

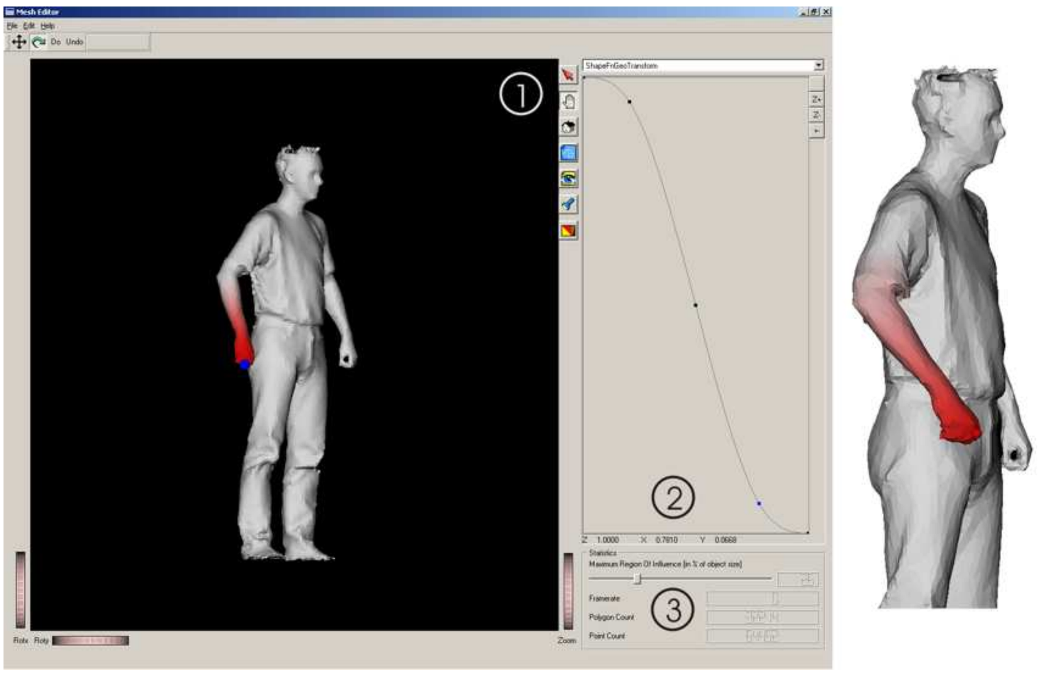


Figure 8.4: *The user interface with an exemplary arm movement. The model was reconstructed from a laser range scan. On the right hand side we see the modified model after applying a translation modification based on geodesic distances. The arm is lifted without affecting the torso. Note the detail preservation leading to the realistic folds of the sleeve.*

In addition to that, interactive editing is not prohibited even for large numbers of handles, as $\mathbf{A}$ is defined during handle and ROI selection, and therefore has to be inverted only once per set of handles. Updating the vertex positions in the deformed region during interactive editing requires only back-substitution which can be performed very efficiently.

Please note that for a single handle vertex $\mathbf{p}_0$, (8.3) reduces to

$$\mathbf{d}(\mathbf{p}) = \alpha(\mathbf{p})\,\mathbf{d}_0$$

with $\mathbf{d}_0 = 1/\alpha(\mathbf{p}_0)\,\mathbf{d}(\mathbf{p}_0)$, leading to a very efficient formulation for the frequent case of single handle vertex edits.

8.2.3 Rotations

As stated above, every transformation can be interpreted as a displacement field, and thus even rotation-like modifications of the object (like turning a person's head) are in theory possible with the above formulation. In general, to achieve satisfying results, this would require a considerable number of consecutive editing steps and/or handle vertices, though. Therefore we use a different kind of constraints for rotational editing operations: Instead of defining *total displacements* for the handle vertices, the user defines *total rotations* $\eta_1, \dots, \eta_k$ with respect to an axis $\mathbf{n}$.

In the current implementation, the rotation axis is simply defined by the screen centre and the viewing direction. Analogously to the displacement field in the translation case in equation (8.1), we define a rotation field

$$\eta : \mathbb{R}^3 \to \mathbb{R}$$

that assigns to every point $\mathbf{p} \in \mathbb{R}^3$ a rotation angle $\eta(\mathbf{p})$ such that the resulting point positions after the modification are given as

$$\mathbf{p}_{new} = \mathbf{R}(\eta(\mathbf{p}_{old}), \mathbf{n})\ \mathbf{p}_{old}, \tag{8.6}$$

where $\mathbf{R}(\eta, \mathbf{n})$ is the rotation matrix that rotates the space by an angle of η about the axis $\mathbf{n}$. As in the translation context, the values of this rotation map are defined at the handle vertices only. For the other vertices the mapping has to be determined as above using

$$\eta(\mathbf{p}) = \sum_{j=1}^{k} \alpha_j(\mathbf{p})\eta_j$$

with *partial angles* η_j and *total angles* $\eta(\mathbf{p})$.

8.3 Parameterisations and Shape Functions

A very simple approach for a weight function could be for example $\alpha_j(\mathbf{p}_i) = \delta_{ij}$ leading to $\mathbf{d}(\mathbf{p}_j) = \mathbf{d}_j$ and $\eta(\mathbf{p}_j) = \eta_j$. This choice of weight functions, however, would give us the generally unsatisfying approach that moves the handle vertices as specified and leaves all other vertices unmodified. Instead, we define the weight functions to be a composition

$$\alpha_j(\mathbf{p}) = \varphi \circ \gamma_j(\mathbf{p})$$

of a *shape function*

$$\varphi : \mathbb{R}^{\geq 0} \to \mathbb{R} \tag{8.7}$$

and a *parametrisation* of the object

$$\gamma_j : \mathbb{R}^3 \to \mathbb{R}^{\geq 0}. \tag{8.8}$$

8.3.1 Parameterisation

The mathematical framework presented in the previous sections is applicable with basically any parametrisation. Handle editing on the other hand suggests some kind of locality in the influence of the handle, such that it seems only natural to parameterise the object in terms of distance from the handle.

Let $\gamma : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^{\geq 0}$ be any distance metric, then

$$\gamma_j(\mathbf{p}) = \gamma(\mathbf{p}_j, \mathbf{p}) \tag{8.9}$$

defines a distance-based parametrisation of the object. The choice of the distance metric is arbitrary in principle, but has a strong influence on the behaviour of the editing method. In order to achieve an intuitive editing behaviour, it is therefore important that γ is chosen such that it defines *intuitive neighbourhoods* on the object.

Although appropriate in some applications and for certain types of objects, choosing Euclidean distances

$$\gamma(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\|$$

as in [S.Yoshizawa et al. 2002] would make it virtually impossible, for instance, to bend a person's index finger without interfering with the other fingers.

On the contrary, *geodesic* distances – the length of the shortest curve between $\mathbf{p}$ and $\mathbf{q}$ on the boundary of the object – define

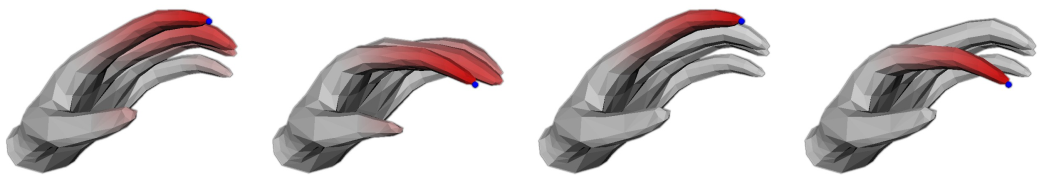


Figure 8.5: *Editing operation with Euclidean (left) and geodesic (right) distances, each before and after the edit. The region of influence is indicated in red colour, the little sphere at the tip of the index finger is the handle that is dragged during the edit. Note how the middle and ring finger are modified together with the index finger in the second left picture.*

neighbourhoods *in the surface* in contrast to the isotropic Euclidean distances and should be used where available. Using definition (8.9), the object is thus parameterised via geodesic distance fields with respect to the handle vertices $\mathbf{p}_j$.

The effect of this choice is illustrated in figure 8.5, which shows an editing operation applied to a simple triangle mesh representation of a hand. Pictures (1) and (3) (counted from left to right) show the original mesh with the region of influence coloured in red. Pictures (2) and (4) show the modified meshes, after an identical edit has been performed, based on Euclidean distances to the left and on geodesic distances to the right. Note how the middle and ring finger have been deformed in picture (2) along with the index figure, whereas in picture (4) the index finger could be modified with no interference with the spatially proximate rest of the geometry.

Raffin et al. [Raffin et al. 2000] propose user-definable *hulls of influence* surrounding the parts of the object that should be affected to achieve results similar to the ones above, but we feel that the geodesic distances provide for a useful *object-inherent* parametrisation for the modification of surfaces, and therefore lead to a more convenient user interface, freeing the user from the need to fit hulls of influence to the specific editing situation (which might be difficult in many cases, e.g. if the fingers are very close in the above example).

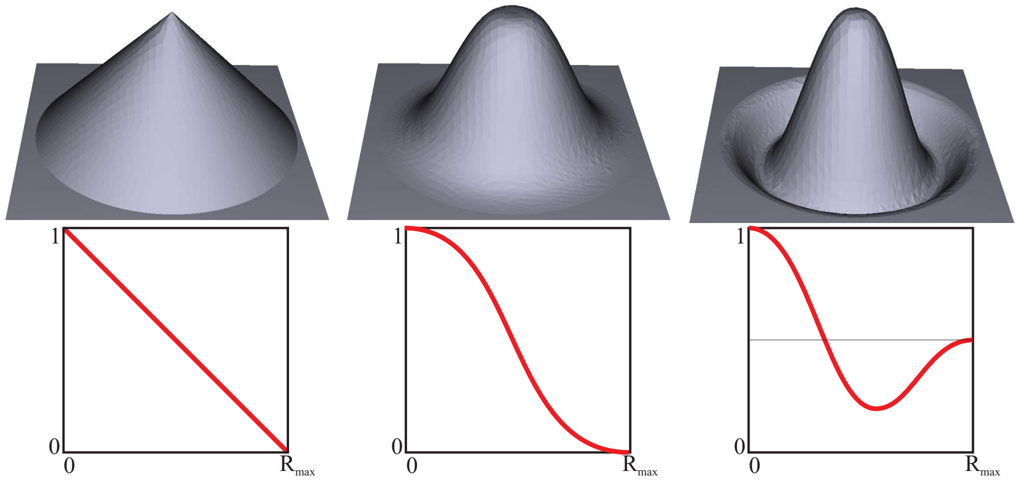


Figure 8.6: *Different shape function settings applied to the same editing operation*

There are numerous approaches for computing geodesic distances for meshes available to date, but for efficiency reasons the current implementation uses the approximative solution of [Novotni & Klein 2002], which allows us to synchronously compute the geodesic distances between the handle vertices and all other vertices. Nevertheless this is computationally nontrivial, but it does not prevent interactive response, since it is performed only when the vertices are selected or deselected, not during dragging. Also, the parametrisation is obviously only required for the ROI and not for the full object. This is not exploited in the current implementation, as here the ROI can still be adjusted interactively even after the edit and is therefore not known a priori. Nevertheless, temporarily restricting the maximal ROI to a certain proportion is a viable optimisation for larger meshes.

8.3.2 Shape Functions

With an appropriate parametrisation at hand, the second component of the weight function α_j, the shape function, has to be defined. For flexibility, the choice of the shape function is left completely at the user's discretion. An illustration of the influence of the shape function on the edit is given in figure 8.6. It is important to mention that the concrete layout of the shape function can be adjusted by the user even after an edit has been specified (just like the ROI as mentioned before). This way, the presented method provides for more flexibility and degrees of freedom than e.g. the trivariate Bernstein polynomials used in FFD methods or the parameterised Gaussian functions used in [S.Yoshizawa et al. 2002].

In many cases, choosing smooth shape functions will be sufficient, but in other cases the user might want to introduce sharp features into the edited area. This can easily be achieved using our approach by employing the appropriate shape function, given that the triangulation of the underlying mesh is adequately fine (see chapter 8.5).

Please note, that with the shape function modelling metaphor, the user is completely free in choosing the shape of the edit, including the creation of sharp creases and even discontinuities. As a consequence, no guarantees of the degree of continuity of the resulting surface can be given. If this is an issue, one can always restrict the space of allowable shape functions α to those which have $(\partial/\partial t)^i \alpha|_0 = 0$ with any desired $i \in \mathbb{N}$.

8.3.3 Separating Handles and Anchors

The method presented so far relies on an object parametrisation with respect to the handle vertices. Whereas this results in an intuitive and easy-to-use tool, there is no theoretical obligation to identify the *handles* used to define the total transformations with the *anchors* used to define the object parametrisation. In some occasions, it might be desirable to parameterise the object with respect to other vertices than the handles. As an example, one can think of turning a person's head (with a rigid head and a smoothly twisted neck) while the rest of the body remains unchanged down from the shoulders (see figure 8.7). In this case it is useful to define the parametrisation with respect to anchor vertices at the top of the head, while the handle vertex can be picked somewhere else on the mesh.

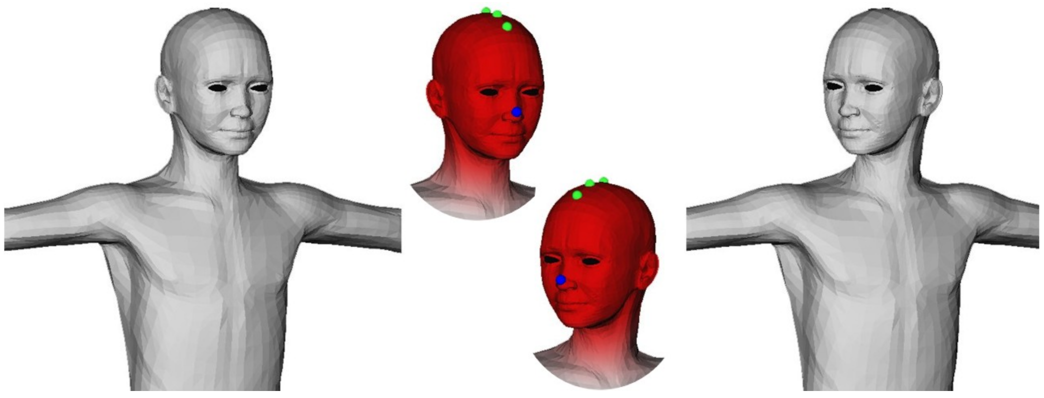


Figure 8.7: *Turning a model's head. As indicated by the red colour, the shoulder region remains fixed, while the head is turned (using the blue handle on the nose). By using multiple (three) anchors (green spheres at the top of the head) we define an anisotropic ROI s.t. the head is turned rigidly, with a smooth changeover at the neck.*

Additionally, separating handles from anchors has another advantage: We can extend this line of thought to a *multiple anchors - single handle*-approach, i.e. the parametrisation of the object is defined with respect to a set of anchor vertices $\mathbf{a}_1^j, \ldots, \mathbf{a}_l^j$ rather than to a single anchor vertex. Thus we are able to define anisotropic distance fields on the object, freeing us from the limitation of rotationally symmetric parameterisations. In the current implementation, the distance field defining the parametrisation corresponding to handle vertex $\mathbf{p}_j$ is then defined for every mesh vertex $\mathbf{p}$ as

$$\gamma_j(\mathbf{p}) = \min_{1 \le i \le l} \gamma(\mathbf{a}_i^j, \mathbf{p}). \tag{8.10}$$

It is known that the iso-values of $\gamma_j(\mathbf{p})$ in equation (8.10) do not form smooth curves. But implementing known techniques from implicit modelling (e.g. following the ideas of Convolution Surfaces from [Bloomenthal 1997]) into our setting is a straightforward extension leading to smooth distance fields.

Severing the ties between handles used for deformation and the definition of its region of influence even further, a straightforward extension to the algorithm presented here would be to let the user determine the influence of any handle by simply *painting* the desired regions of the object. A deeper saturation could then be interpreted as a stronger influence etc. (just like in the illustrations throughout this chapter).

8.4 Mesh Forging Process

The basic idea of mesh forging is to add an *occluder* (the anvil) to the editing space in form of a force field, thereby replicating the *Precise Contact Modelling* (PCM) methodology [Gascuel 1993] in the context of mesh editing. Here, the occluder field controllably superposes (and thereby modifies) the transformation applied to the vertices of the mesh (cf. figure 8.8 for an example).

In implicit modelling, contact situations between two surfaces

$$S_i = \{\mathbf{p} \in \mathbb{R} \mid f_i(\mathbf{p}) = c\}, \qquad i \in \{1, 2\}$$

are easily detected by checking for points $\mathbf{p}$ satisfying both $f_1(\mathbf{p}) \leq c$ and $f_2(\mathbf{p}) \leq c$. For these points in the interpenetration region, a *compression term* is added to the field function f_i. If only S_1 is deformable and S_2 rigid, $f_1(\mathbf{p})$ is replaced by

$$c + (c - f_2(\mathbf{p}))$$

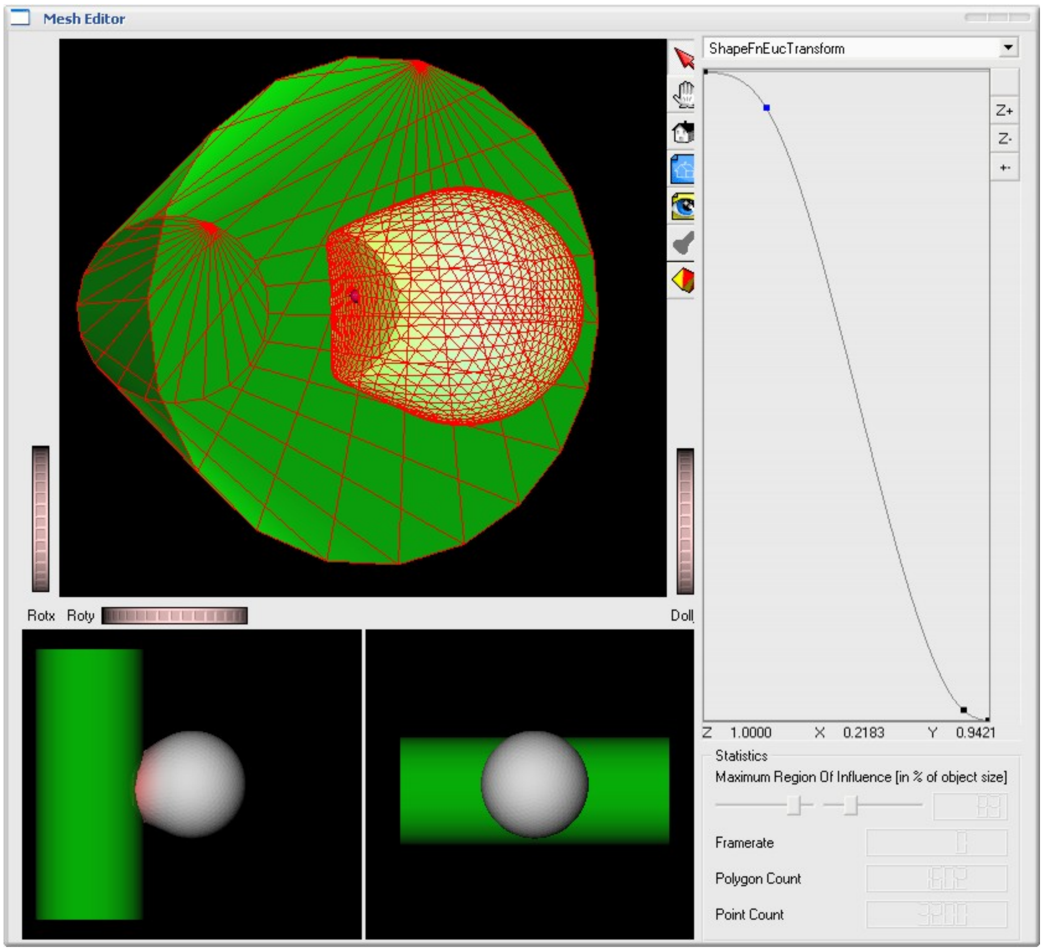
for all $\mathbf{p}$ in the interpenetration region.
In order to mimic volume preservation in the contact regions, a *dilation term* $b(\mathbf{p})$ is added for points $\mathbf{p}$ in the propagation region

$$\{\mathbf{p} \in \mathbb{R} \mid \tilde{c} \geq f_2(\mathbf{p}) > c\}$$

with some constant $\tilde{c}$. For an in-depth description of the PCM methodology, see [Cani-Gascuel & Desbrun 1997; Opalach & Cani-Gascuel 1997].

8.4.1 The Algorithm

Suppose that in a mesh editing environment, we have a vertex $\mathbf{p} \in \mathbb{R}^3$ in the mesh and, defined by some modelling operation,



ShapeFnEucTransform

Z 1.0000 X 0.2183 Y 0.9421

Figure 8.8: *Example of a mesh forging operation. A vertex of the editing object (the grey coloured ball) is picked and dragged. The occluder (the green cylinder) induces a force field that superposes the displacement field and drives the transformed vertices around it.*

a total displacement $\mathbf{d}(\mathbf{p})$ for this vertex. Suppose further that there is an occluder $\mathcal{O}$ in the scene. The editing operation should now go exactly as described hitherto except where in conflict with the occluder object. Mesh forging therefore requires the detection of collisions between the edited regions of the manipuland with the occluder, which can be easily performed for an occluder represented in implicit form. In order to detect collisions even for

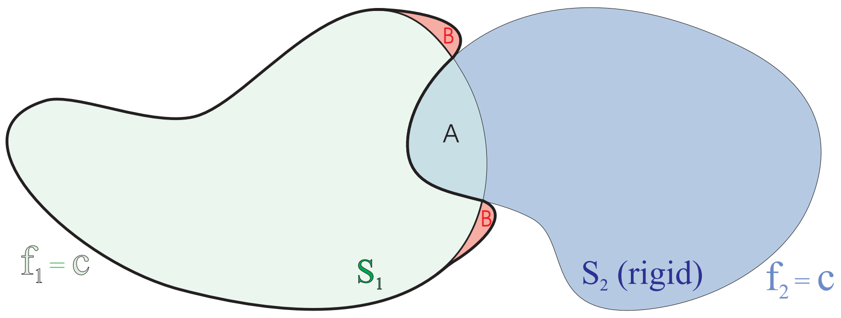


Figure 8.9: *Precise contact modelling (PCM) for implicit surfaces. In contact situations, a surfaces S_1 and S_2 are defined by modifying the defining implicit functions $f_1 = c$ and (or) $f_2 = c$, respectively, in the* interpenetration region *(A) and optionally in the so-called* dilation region *(B).*

larger edits, it makes sense to first subdivide $\mathbf{d}(\mathbf{p})$ – otherwise it would be possible to move undetectedly *through* the occluder or through occluder details. The algorithm then leads to the following sequence of transformations (see figure 8.10):

$$\begin{array}{ccccl}
\mathbf{p} & \mapsto & \mathbf{p}^1 & = & \mathbf{p} + \frac{1}{N}\,\mathbf{d}(\mathbf{p}) + \mathbf{o}\left(\mathbf{p} + \frac{1}{N}\,\mathbf{d}(\mathbf{p})\right) \\
\mathbf{p}^1 & \mapsto & \mathbf{p}^2 & = & \mathbf{p}^1 + \frac{1}{N}\,\mathbf{d}(\mathbf{p}) + \mathbf{o}\left(\mathbf{p}^1 + \frac{1}{N}\,\mathbf{d}(\mathbf{p})\right) \\
 & \vdots & & & \\
\mathbf{p}^{N-1} & \mapsto & \mathbf{p}^N & = & \mathbf{p}^{N-1} + \frac{1}{N}\,\mathbf{d}(\mathbf{p}) + \mathbf{o}\left(\mathbf{p}^{N-1} + \frac{1}{N}\,\mathbf{d}(\mathbf{p})\right)
\end{array}$$

Here, $\mathbf{o}(\mathbf{p})$ is the occluder field that corresponds to the compression term in the implicit modelling context. To describe the editing process in general, we get the following recursive algorithm:

$$\mathbf{p}^{i-1} \mapsto \mathbf{p}^i = \mathbf{p}^{i-1} + \frac{1}{N}\,\mathbf{d}(\mathbf{p}) + \mathbf{o}\left(\mathbf{p}^{i-1} + \frac{1}{N}\,\mathbf{d}(\mathbf{p})\right) \tag{8.11}$$

Our new transformation equation can then be written as

$$\mathbf{p} \mapsto \mathbf{p} + \mathbf{d}(\mathbf{p}) + \sum_{i=1}^{N} \mathbf{o}(\mathbf{p}_i). \tag{8.12}$$

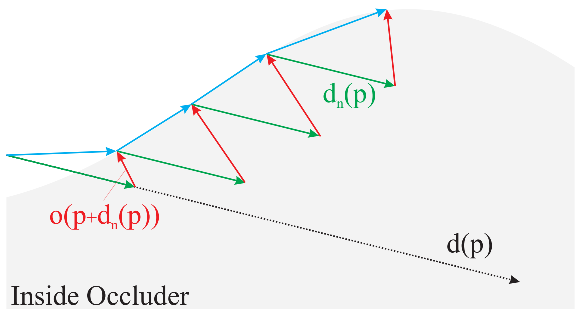


Figure 8.10: *Successive occluder influence. The stippled vector indicates the vertex transformation* $\mathbf{d}(\mathbf{p})$ *as defined by the editing operation, the green vectors represent the n-th part* $\mathbf{d}_n(\mathbf{p}) = 1/n\ \mathbf{d}(\mathbf{p})$ *of this transformation. The red vectors indicate the repelling force due to the occluder field. Note how the considered vertex* $\mathbf{p}$ *moves along the boundary of the occluder leading to the expected editing behaviour.*

Please note that (8.12) becomes

$$\mathbf{p} \mapsto \mathbf{p} + \mathbf{d}(\mathbf{p}) + \int_{\mathcal{P}} \mathbf{o}(\mathbf{p}(s))ds \tag{8.13}$$

for $N \to \infty$, which is a natural generalisation of equation (8.2). Here, $\mathcal{P}$ is the path through the occluder field from $\mathbf{p}$ to $\mathbf{p}+\mathbf{d}(\mathbf{p})$.

Although the above formulation would allow for a complete tracking of the editing path as indicated by the mouse movement, the displacement field $\mathbf{d}$ is still evaluated at $\mathbf{p}$ only. The reasoning behind this is that a complete tracking of the mouse movement turned out to be rather a hindrance than helpful during editing.

8.4.2 Defining the Occluder Field

For the efficient detection of contact situations we define the occluder implicitly as a signed distance vector field, i.e. in addition to the signed distance

$$\begin{array}{rccl} \delta: & \mathbb{R}^3 & \to & \mathbb{R} \\ & \mathbf{p} & \mapsto & \text{signed distance of } \mathbf{p} \\ & & & \text{to occluder surface,} \end{array} \tag{8.14}$$

we store for every point $\mathbf{p} \in \mathbb{R}^3$ also the direction

$$\begin{array}{rccl} \Delta : & \mathbb{R}^3 & \to & \mathbb{R}^3 \\ & \mathbf{p} & \mapsto & (\mathbf{c} - \mathbf{p})/\|\mathbf{c} - \mathbf{p}\|, \end{array} \tag{8.15}$$

to the closest point $\mathbf{c}$ on the occluder surface. We discretionarily choose $\delta(\mathbf{p}) < 0$ iff $\mathbf{p}$ is inside the occluder.

The well-known Adaptively Sampled Distance Fields (ADFs) [Frisken et al. 2000] are well-suited for this purpose here since the sample density of the ADF can be used as a hint for the sampling distance for the editing paths. We propose using the voxel width as a local path sampling rate. This inherently allows for feature detection in the occluder field.

Although the current implementation makes use of analytically defined occluders, a future toolbox will contain a set of predefined signed distance fields. The distance fields to user-defined occluders have to be calculated in a preprocessing step. However, this does not prevent user interaction with the occluder: Resizing, translating, rotating are all trivially available without changing the actual values in the distance field. Evaluating the *transformed* distance field at a position $\mathbf{p}$ simply requires the evaluation of the original distance field after applying the inverse transformation to $\mathbf{p}$. Obviously, occluders given a priori in implicit form can be incorporated as-is.

Having access to the distance values and to the closest point on the occluder surface at any position in space, all the ingredients are available to formulate the occluder force field. We define

$$\begin{array}{rccl} \mathbf{o} : & \mathbb{R}^3 \times \mathbb{R}^3 & \to & \mathbb{R}^3 \\ & (\mathbf{p}, \mathbf{d}(\mathbf{p})) & \mapsto & \mathbf{o}(\mathbf{p}, \mathbf{d}(\mathbf{p})) \end{array} \tag{8.16}$$

as follows:

$$\mathbf{o}(\mathbf{p}, \mathbf{d}(\mathbf{p})) = -\psi(\delta(\mathbf{p} + \mathbf{d}(\mathbf{p}))) \cdot \Delta(\mathbf{p} + \mathbf{d}(\mathbf{p}))$$

where $\psi : \mathbb{R}^3 \to \mathbb{R}$ is an *influence function* that can be thought of as a kind of shape function for the occluder and that determines the effective impact of the occluder field. Depending on the actual

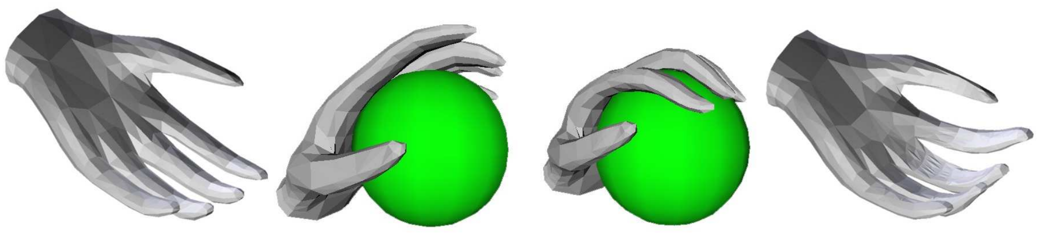


Figure 8.11: *Modelling a hand taking grip on a ball. The fingers of the hand are transformed by a simple drag on the finger tips. The force field induced by the occluder causes the fingers to be shaped around the ball instead of intruding into it.* Influence functions *prevent the fingers from flattening.*

editing circumstances, different influence functions are appropriate, e.g.

$$\psi(\delta(\mathbf{p})) = \begin{cases} \delta(\mathbf{p}) & : \quad \delta(\mathbf{p}) \leq 0 \\ \exp(-\delta^2(\mathbf{p})) & : \quad \delta(\mathbf{p}) > 0. \end{cases} \tag{8.17}$$

This influence function guarantees that vertices penetrating the occluder are transferred to the occluder surface, regardless of the underlying editing operation, and vertices coming close to the occluder surface but not penetrating it are also repelled. This prevents (to some extent) the fingers from flattening in figure 8.11.

The rationale behind formulating $\mathbf{o} = \mathbf{o}(\mathbf{p}, \mathbf{d}(\mathbf{p}))$ instead of $\mathbf{o} = \mathbf{o}(\mathbf{p})$, i.e. making the occluder field not only dependent on the *locus* $\mathbf{p}$ but also on the editing *direction* $\mathbf{d}(\mathbf{p})$ is that this leads to a more flexible approach. The most significant benefit from this formulation is that we are able to assure that the occluder has no bigger effect than the originating displacement and therefore to restrict the occluder influence to the editing region of influence. This can easily be done by including $\|\mathbf{d}(\mathbf{p})\|$ as a factor into equation (8.17).

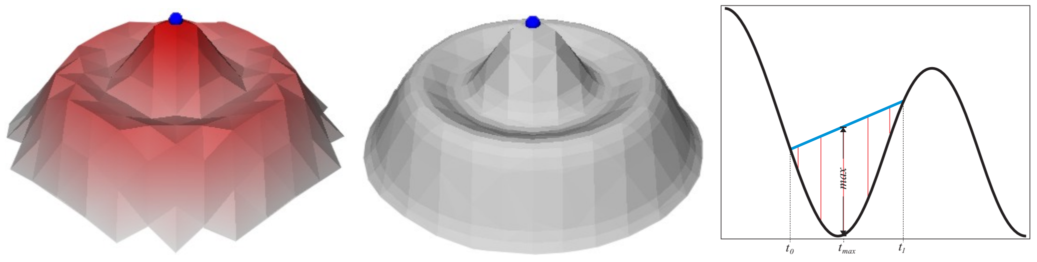


Figure 8.12: *Left: For some edits, the triangles might be too large to represent fine details as defined by the shape function. Middle: Result after refinement. Right: The error induced by applying a transform to a polygonal mesh as specified by the shape function* without *refining the mesh corresponds to the error induced by linearly interpolating the shape function.*

8.5 Adaptive Refinement

Editing operations change the geometric properties of the underlying surface. In particular, edits are likely to add small details that might not be representable by the current triangulation, such that an adaptive refinement has to be performed, e.g. as proposed in [Greissmair & Purgathofer 1989] or [Gain & Dodgson 1999]. In these approaches, local curvature information (by midpoint subdivision or by vertex normal deviation, resp.) is used to decide if further refinement is required.

In addition to this general refinement technique, shape function editing allows exploiting the shape function information to decide if and *where* edges have to be subdivided. This way, even sharp feature edits can be incorporated. In order to determine refinement candidates, consider the error induced by linearly interpolating the shape function between adjacent vertices (see figure 8.12, right). For a handle vertex $\mathbf{p}$ and an edge $(\mathbf{v}_0, \mathbf{v}_1)$ let

$$\epsilon^{\mathbf{p}}_{\mathbf{v}_0,\mathbf{v}_1} = \max_{t_0 \le t \le t_1} \left| \varphi(t) - \varphi(t_0) + \frac{t - t_0}{t_1 - t_0} \left(\varphi(t_1) - \varphi(t_0)\right) \right| \qquad (8.18)$$

with $t_0 = \gamma(\mathbf{p}, \mathbf{v}_0)$, $t_1 = \gamma(\mathbf{p}, \mathbf{v}_1)$ and φ and γ as defined in (8.7) and (8.8) respectively. For multiple handle vertices $\mathbf{p}_1, \ldots, \mathbf{p}_k$ we define

$$\epsilon_{\mathbf{v}_0,\mathbf{v}_1} = \max_{\mathbf{p}=\mathbf{p}_1,\ldots,\mathbf{p}_k} \epsilon^{\mathbf{p}}_{\mathbf{v}_0,\mathbf{v}_1}.$$

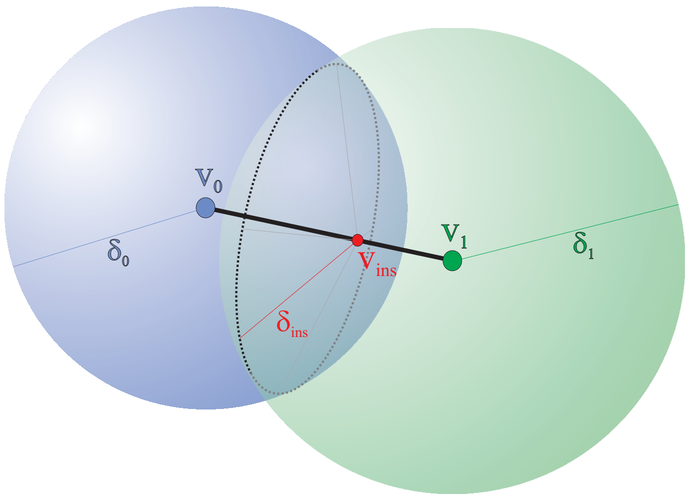


Figure 8.13: *Approximation of the geodesic distance for a new vertex* $\mathbf{v}_{ins}$ *inserted on the edge between* $\mathbf{v}_0$ *and* $\mathbf{v}_1$*.* δ_{ins} *is approximated as the distance to the intersecting circle (dashed) between the* δ_0*-sphere around* $\mathbf{v}_0$ *(blue) and the* δ_1*-sphere around* $\mathbf{v}_1$ *(green).*

Edges are subdivided if ϵ exceeds a user-controllable threshold.

Let t_{ins} be the parameter in $[t_0, t_1]$ for which the right hand side in (8.18) becomes maximal. A new vertex will then be inserted into this edge at the *temporary* position

$$\mathbf{v}_{ins} = \mathbf{v_0} + \frac{t_{ins}}{t_1 - t_0} (\mathbf{v}_1 - \mathbf{v}_0) .$$

After the insertion of new vertices, the new edges are checked if further refinement is required. Finally, the new positions for all vertices in the mesh are calculated. In this step we take advantage of the fact that we can easily approximate the geodesic distance for every new vertex $\mathbf{v}_{ins}$ on the edge $(\mathbf{v}_0, \mathbf{v}_1)$ from the distance values of the two adjacent vertices. To this end, let δ_0 and δ_1 be the distances of the vertices $\mathbf{v}_0$ and $\mathbf{v}_1$ to the handle respectively. We compute a virtual origin for the distance calculation on the intersection of the two spheres with radii δ_0 and δ_1 and centres $\mathbf{v}_0$ and $\mathbf{v}_1$ (see figure 8.13). Since the intersecting circle is orthogonal to the edge $(\mathbf{v}_0, \mathbf{v}_1)$, every point on it has the same distance to

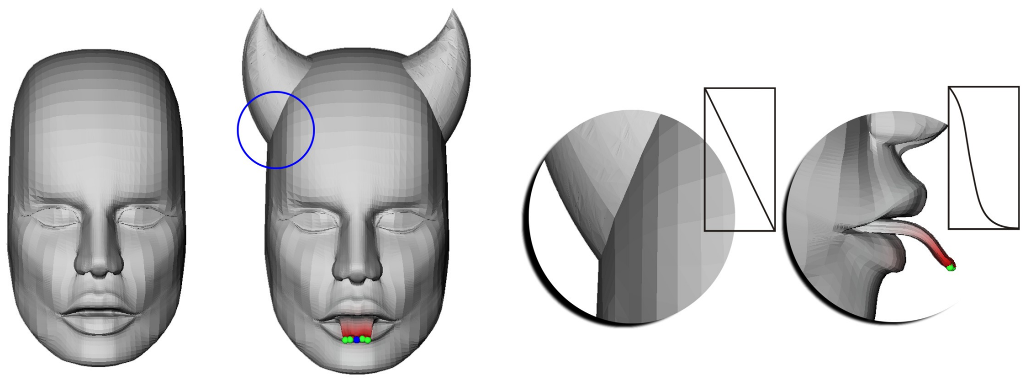


Figure 8.14: *Sharp features can be modelled because the mesh is adaptively refined according to an edit applied to the mesh.*

$\mathbf{v}_{max}$ which is used as the desired approximation of the geodesic distance in $\mathbf{v}_{ins}$. For further details see [Novotni & Klein 2002].

Whereas the above refinement strategy allows for sharp feature editing by recursively subdividing edges where indicated by the shape function, further refinement might be required to model contact situations, as can be seen in the rightmost picture in figure 8.11.
Moreover, this is a refinement strategy based on the shape function edit only. It does not take into account the underlying geometric properties of the object in the editing region. Therefore, even in cases in which the edit actually reduces high frequencies on the mesh, new vertices might be inserted.

8.6 Editing Examples

In addition to the modelling examples given so far in this chapter, this section will present several further modelling samples to prove the feasibility and potential of the presented editing paradigm.

Figure 8.15 shows how the bunny's ears can be transformed with one single editing operation consisting of as few as four editing steps. Firstly, the tips of the two ears are selected as handles; secondly, a preliminary choice for the ROI of the edit is made (this can always be changed at later stages of the edit, the preliminary choice only improves the visual feedback during the following steps). As stated before, the current implementation derives the rotation axis from the screen centre and the viewing direction. Hence, the bunny is positioned accordingly and the tips of the ears are dragged by the user to the desired position (figure 8.15, second image, bottom row). After the transformation for the tips of the ears has been specified, the shape function can be modified interactively in order to achieve a realistic look of the ears (third image, bottom row). Please note that – as was also pointed out by Botsch et al. [2006] – it would have been very hard to achieve this result using the translation scheme only, as can be seen in figure 8.15 (right image, bottom row).

Figure 8.7 is an example for the use of the anisotropic rotational editing scheme. The rotation axis was chosen in this example approximately parallel to the spine. Note that only the head has turned, the shoulder region remained fixed. In order to have the head remain rigid while the neck is twisted, a small number of anchor vertices has been chosen on the top of the head to define the region of influence, and a single handle vertex was picked on the nose for transformation determination. The multiple anchor vertices are necessary because otherwise it would have been difficult to adjust the region of influence such that only the neck is twisted (the head is not perfectly round, so choosing only one anchor vertex at the top will not lead to a satisfying region of influence behaviour – with just one anchor vertex either the shoulder region would have been influenced or ”outer regions” as the chin would not have moved rigidly with the rest of the head).

The option of choosing multiple anchor vertices is also essential for editing operations as depicted in 8.14 (to the right). This figure also illustrates the adaptive refinement method, by which it is possible to model sharp features even in sparsely triangulated regions of the mesh. However, our refinement strategy is based on the shape function edit only. It does not take into account the underlying geometric properties of the object in the editing region. As mentioned above, it might be worth looking also into the reduction of mesh complexity where appropriate, while maintaining

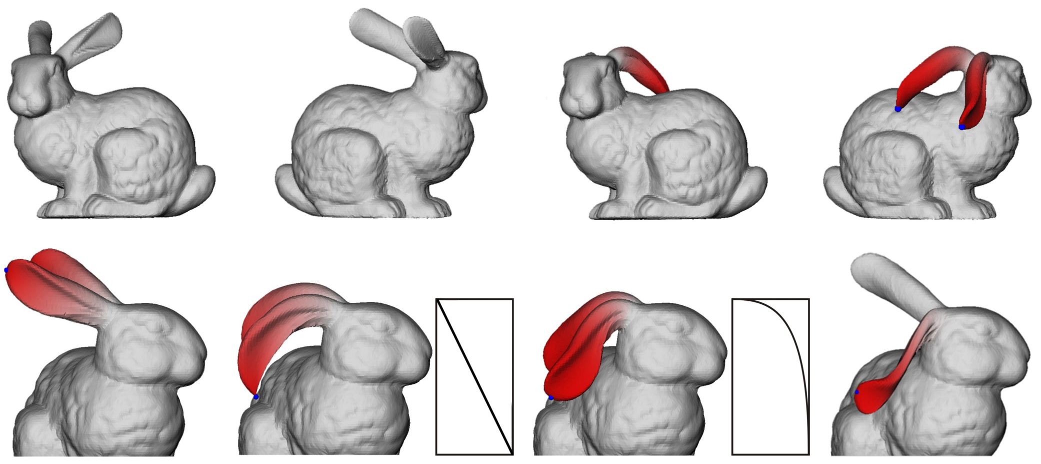


Figure 8.15: *Application of the rotational editing scheme. The tips of the bunny's ears are picked and dragged, while not only the region of influence (indicated by the red colored area) but also the shape of the edit can interactively be modified and adjusted using the shape function (boxes in the bottom row) until the impression is visually satisfying. The bottom right picture shows the corresponding edit using the translational scheme with a slightly adjusted shape function to produce a smooth changeover at the bunny's head.*

the efficiency of analysing only the shape function. This has not been in the scope of the present work, though. On the other hand, reducing the mesh complexity in flat surface regions and improving the mesh quality in general can always be achieved with the aid of any of the numerous remeshing techniques available, given that the mesh contains all the required detail, as ascertained with the above refinement method.

Figure 8.16 (left) illustrates the benefit of being able to interactively modifying the shape function *and* the region of influence. For cases like this, where the user has a very well-defined vision of how the edit should look like (in this instance the bend should be where the elbow is), it is sometimes hard to determine the ROI and the specific shape of the edit beforehand. With the approach presented in this chapter, the user can modify both aspects of the edit *afterwards*. Figure 8.16 (right) is a further example, where the parametrisation based on geodesic distances as an object-inherent

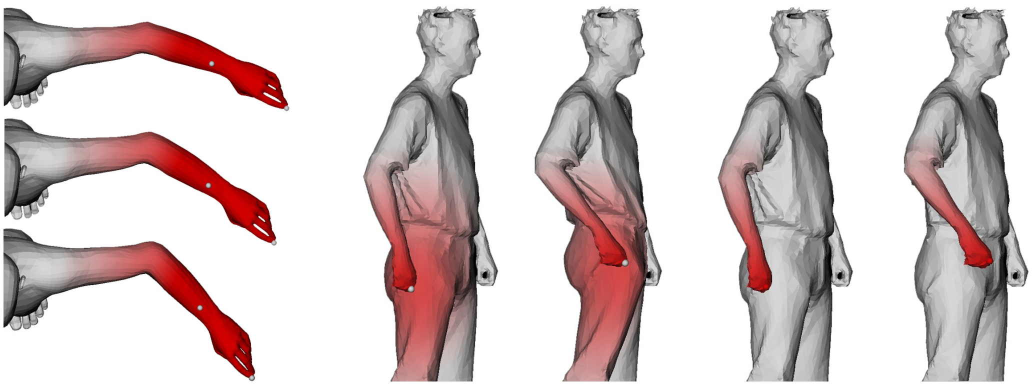


Figure 8.16: *Modelling an arm bend. The images to the right give another example where geodesic parameterisations perform superior to Euclidean ones.*

parametrisation is far superior to a parametrisation based on Euclidean distances.

In many cases picking one or more vertices on the mesh and dragging them to the desired position is sufficient and leads to satisfying results. In order to give users further flexibility, it is possible with the presented approach to to specify the parametrisation of the object *independently* from the handles (cf. section 8.3.3). Among other benefits, this simplifies the editing operation in rotation cases where the distance field is defined on (or close to) the rotation axis. In these cases, it is convenient to grab a different point on the object to specify the rotation angles.

The present mesh editing approach can not only be used for animation but also for creative modelling purposes as it can be seen from figure 8.17. Starting from a sphere, a complete teapot is created within minutes. Note how the shape function can be used to model details on the teapot's handle with a rotational edit.

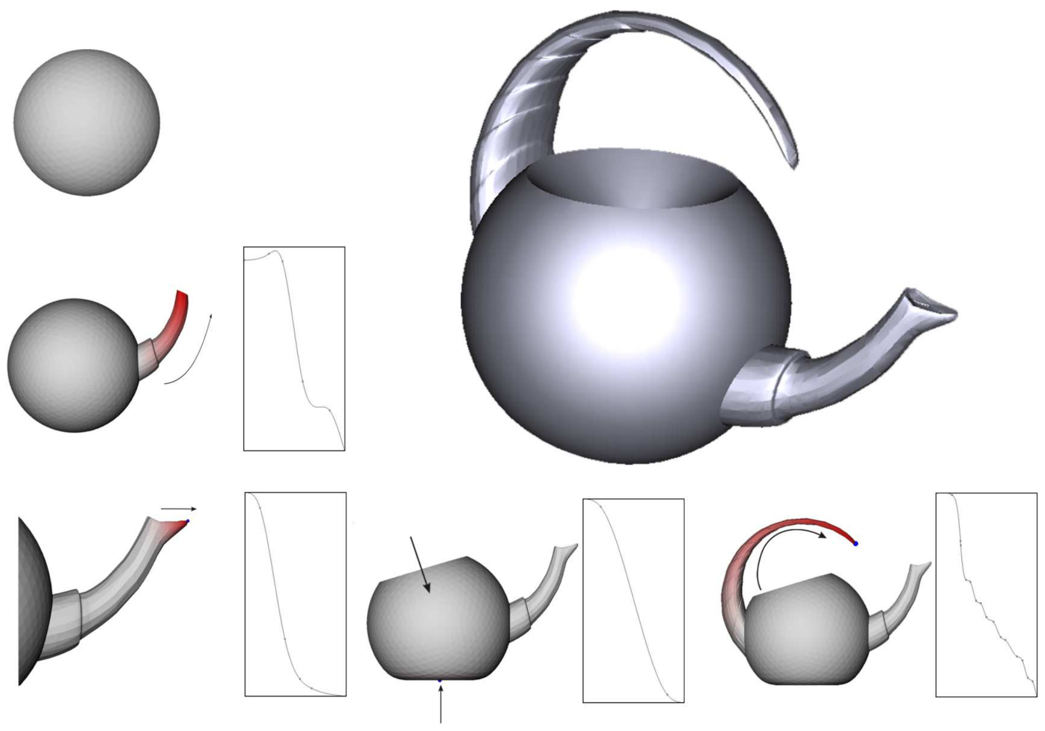


Figure 8.17: *Creating a teapot (top right) from a primitive (top left) with just a few editing operations. The basic editing operations are depicted together with the corresponding shape functions (left and bottom). The arrows indicate the modification applied to the handles. Note how the shape function can not only be used to adjust the overall shape of the edit but also to add details to the model (see the teapot's handle in the last editing step and in the final result).*

Chapter 9

Modelling for Surface Inpainting

Chapter 7 of this thesis dealt with the reconstruction and repair of digital models – a task that is frequently encountered in various applications, but in particular in the fields of cultural heritage and archaeology. Here, one is often faced with models that are not only incomplete in the sense of imperfect or fragmentary data due to a deficient data acquisition process, but also because the data *source* may already be incomplete. Faithfully digitised models are often to be restored in order to visualise the object in its original state, reversing the effects of aging or decay. Where the automatic surface completion from chapter 7 targeted a repair on the basis of the acquired data without additional information, this chapter will discuss how a user's expertise and creativity (where desired) can be incorporated into such an automatic completion system. To this end, intuitive free-form modelling techniques are combined with automatic 3D surface completion. Thus, a user's expertise can be included into the surface completion process, which, in turn, reconstructs the required surface detail in the modelled region and thereby frees the user from the need to model every last detail manually.

9.1 Introduction

Although the faithful reconstruction of the scanned objects for visualisation and presentation purposes obviously is generally the first and most important step of all 3D data acquisition projects, the exploitation of the acquired data does not end there. Recreating times passed is an important aspect of nowadays historic research and not least in the entertainment industries and education. And in this context, the modelling and transformation also of scanned models is becoming an important task, just as it has traditionally been for other data sources such as CAD. However, for this kind of application, the requirements of an acceptable modelling tool are in several respects different from those in other application fields.

Before this background, this chapter introduces a novel 3D surface editing approach that to a certain extent resembles a stone mason's approach to restore a historic artefact that has been damaged be external influences as weather or wear. After removing the defective part of the object, a roughly pre-shaped *template* is inserted into the defective region and then modelled to recreate the original shape of the object. However, modelling every last detail manually would require not only a considerable amount of expertise but would also be extremely tedious and time consuming. Therefore, we will describe a modelling framework that combines free-form modelling (to define the basic layout of the patch to be recreated) with automatic shape completion that uses this basic layout as a guidance and automatically transfers suitable details from other regions of the object.

The motivation behind this two-fold approach is that neither free-form modelling of the defective area nor removing it and automatically filling the emerging hole is generally a feasible solution to the problem of incomplete or damaged surfaces in 3D. As will be demonstrated in the following, however, combining these two complementary *modi operandi* leads to a very powerful modelling methodology, that is capable of including a user's expertise into the otherwise automatic surface completion.

9.2 Related Work

In chapter 7 it was argued that the acquisition of real-life 3D models using laser-range scanners, structured light or even tactile sensors almost inevitably leads – due to occlusion, the object's material properties or spatial constraints during recording – to incomplete surfaces. In tradition of successful image completion approaches, chapter 7 introduced an automatic surface completion method that is able also to reproduce fine surface detail wherever such detail can be presupposed from the context of the hole.

This approach, however, shares with any example-based approach two fundamental and unavoidable restrictions:

- Example-based approaches search and find appropriate image or surface patches based on an analysis of the context of the target region. It can therefore deliver plausible results only if the context does contain significant indications of the shape of the object in the target region.

This restriction is a strong limitation for any automatic surface completion method. A human observer – in particular with the experience and expertise of trained historians – in many cases is able to solve ambiguities and uncertainties that a completely automatic approach may suffer from. The new approach presented here exploits this by allowing the user to transfer this knowledge into the system with the help of an intuitive editing paradigm.

- Example-based approaches search and find appropriate image or surface patches in a so-called candidate set, that is made up of fragments of the image or object itself or of objects accessible in some kind of database. It can therefore deliver plausible results only if the candidate set does contain appropriate fragments that fit into the target region. *Singular* features cannot be reconstructed.

In the context of coherence and similarity detection, it is important to recall from section 7.5 that surface features in 3D objects might be distributed over several different scales. As a consequence, similarity relations on an object can be very different for features present in even only one target region. Therefore, the missing surface features have to be reconstructed *per scale.* This is of particular importance here, as the details in a missing surface region should be reconstructed up to a certain scale only, whereas larger (coarser) scale features are modelled by the user and should be respected by the automatic completion approach.

The hierarchical completion algorithm identifies best fitting candidate fragments based on so-called 2-layer descriptors, where the first layer captures the geometry of the fragment up to a scale that is defined by the fragment size, and the second layer captures the geometry on a coarser level. Thus, the geometry information of *an already completed* coarser level can be included in the identification of appropriate candidates. This way, fragment sizes can be chosen to correspond well to the scale of the features to be reconstructed, whereas without guidance surface, small fragment sizes would typically result in surface patches growing from the hole border to the inside of the hole independently and possibly without meeting, delivering unacceptable shapes.

As was discussed in section 7.5, however, creating an initial hypothesis of the missing surface region is a challenge, that was approached by creating a smooth surface patch from the vertices bordering the hole region. While this procedure is justified because the missing surface patch can be assumed to be smooth on coarse levels, the option to include the user's expertise into the guidance surfaces has not been considered hitherto. It is therefore impossible to reconstruct features that are geometrically unindicated by the context. In a nutshell, the basic idea is now to let the user edit surface templates in the hole region and thereby guide the automatic completion via interactive surface modelling.

In a sense, this approach is conceptually similar to a 2D image completion approach that was recently presented by Sun et al. [2005]. In their approach users are enabled to draw lines in an image to indicate the large scale layout of features that would otherwise be ambiguous or could not be reconstructed for other reasons (*Manual Structure Propagation*). Likewise, we let users *sketch* the basic geometric layout of a missing surface region, via template insertion and free-form modelling, before finally, automatic surface completion is responsible for the recreation of the fine detail structures in the formerly defective surface region.

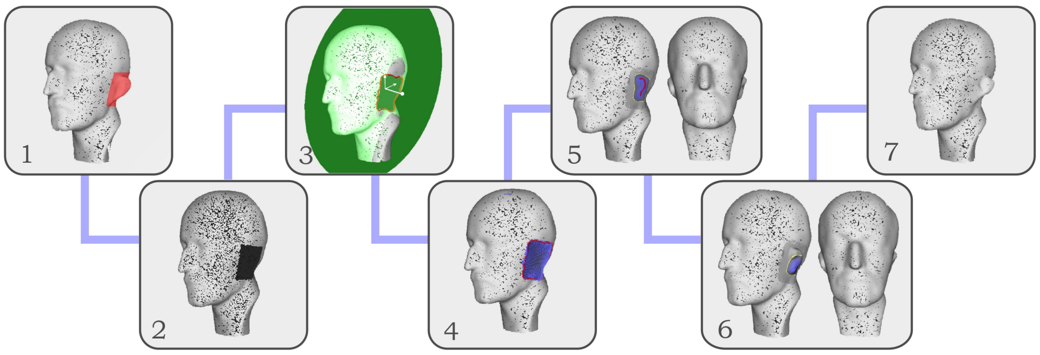


Figure 9.1: *The Modelling Workflow: Marking the defective area (1), incomplete surface (2), (generic) template alignment (3), warped template (4), template modelling (two steps, 5 & 6), result after automatic completion (7).*

9.3 Framework Overview

The basic layout of the novel algorithm is illustrated in figure 9.1 which shows the overall workflow of the *modelling pipeline*: The framework expects as input an unstructured point cloud $\mathcal{P} \subseteq \mathbb{R}^3$, as usual approximating a 2-manifold surface. The first step of the algorithm is to compute a scale-space approximation $\mathcal{P}^0, \ldots, \mathcal{P}^L$ of the given point cloud, consisting of ever coarser approximations of the underlying surface. In this object representation, the user specifies (using standard paintbrush techniques) the defective area to be repaired (1 & 2). In a second step a template is introduced into the framework, either by inserting a generic template (e.g. a plane) or by selecting a part of the object under consideration itself.

This template is then roughly aligned to the defective object region (3). In order to establish a continuous transition between original and template, the border of the defective region is detected, and corresponding line segments are automatically found on the template. In an automatic warping step, the template is non-rigidly transformed mapping these lines onto each other (4). These lines also serve as constraints for the ensuing modelling phase and define the maximal ROI of the modelling operations to be applied later on. Thus guaranteeing that the original model remains unchanged, the inserted template can be modelled to define the basic geometric layout of the shape to be reconstructed (5 & 6).

Of course, in this phase the user can model the inserted template to any desired level of detail; typically, however, only a few modelling operations are necessary, roughly indicating the shape of the region to be recreated. These indications are then the key ingredients in the following surface completion phase where the original defective surface is iteratively replaced by the new synthetic surface patch, recreating also its fine geometric detail properties (7). To this end, the target region is analysed and suitable candidate fragments (as defined in section 7.3) are detected, copied and transferred to the defective surface region. In this latter phase, the scale space representation including the modelled template surface is exploited as guidance to identify appropriate candidates, and directs the fragment insertion spatially.

The following sections will discuss each of the respective phases in detail.

9.4 Template Insertion

The basic idea for the modelling phase is similar to the well-known principle of multi-resolution modelling: On smooth scales, coarse edits are performed by the user, whereas handling the details is left to the algorithm. Unfortunately, this principle is not viable as such in the present application setting, as the geometric layout of the defective region might be very different – even on coarse scales – to the desired surface (see figure 9.2). It might therefore be difficult and require considerable modelling effort to transform this defective surface into the desired shape. Hence, the first step of our algorithm is to simply remove the defective part of the given object.

After this, however, a user's expertise can only be included into the surface completion automatism via modelling if some kind of surface to be modelled exists in the missing surface region in the first place. While this could be achieved using any of the

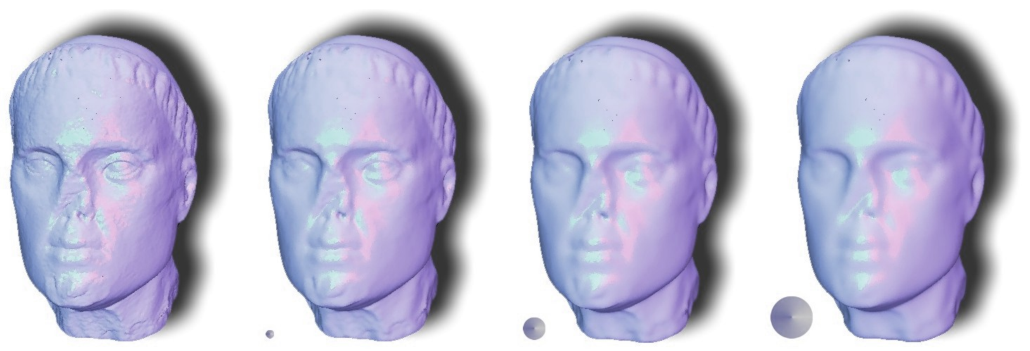


Figure 9.2: *Scale-space representation of the MaleHead-model. The little discs indicate the size of the smoothing kernel used to derive the coarser scales. Note the prominent defective feature in the nose region, that still is present after considerable filtering (far right).*

aforementioned smooth hole filling approaches, another approach turned out to be much simpler and more flexible to fulfill the given requirements. The main idea is to incorporate template surface patches into the hole region. For maximum flexibility, these templates can either be generically constructed, such as planes, cylinders or spheres, or be selected from other objects (or parts thereof). This way, the user is given a much more suitable prior to start from than the defective surface.

9.4.1 Non-Rigid Alignment

On the other hand, the template needs now to be fitted to the scanned model to produce a continuous transition between original and template surface. To this end, the template has to be positioned and non-rigidly deformed to match the surrounding surface.

Let $\mathcal{B} \subset \mathcal{P}$ be the set of boundary points, i.e. of points in close vicinity to the hole, and let the template be represented by a set $\mathcal{T} \subset \mathbb{R}^3$ of points.[1] After the user has roughly pre-aligned the template with the hole, a few steps of what could be under-

[1]Boundary points are either known by construction or detected with the technique described in chapter 6.

stood as a constrained-domain ICP [Besl & McKay 1992] are performed, i.e. we automatically compute corresponding point pairs $(\mathbf{b}, \mathbf{t}_b) \in \mathcal{B} \times \mathcal{T}$, apply the minimising transformation and iterate. Alternatively to manually positioning the template, the user can also specify a small number of explicit correspondences, and the transformation minimising their distances is applied.

This way, the template is co-aligned with the hole boundary, but does not generally constitute an exact match due to the potentially different geometric layout of the template with respect to the defective model. The template therefore needs to be deformed. One straightforward approach would be to find a smooth morphing function by variational optimisation as e.g. described in [Allen et al. 2003] and [Pauly et al. 2005]. However, minimising the penalty functionals is a computationally demanding and time-intensive process already for meshes and more severely for the much more densely sampled point sets.

Instead, for conceptual simplicity, we derive a conforming transformation of the template as an automatic morph on the basis of the modelling methodology from the previous chapter. Here, the constraints are defined by the corresponding point pairs $(\mathbf{b}, \mathbf{t}_b)$, i.e. equation (8.14) becomes

$$\mathbf{d}(\mathbf{t}_b) = \mathbf{b} - \mathbf{t}_b. \tag{9.1}$$

In order to ensure a continuous transition between template and original surface, however, a considerable number of point constraints is generally required.

As inverting $\mathbf{A}$ for a dense set of constraints leads to numerical instabilities (even though analytically the matrix inverse is well-defined if only the constraints are smooth enough) we therefore use *generalised constraints* rather than the point-constraints used in [Borrel & Rappoport 1994].

9.4.2 Generalised Constraints

In the previous chapter, as well as in [Borrel & Rappoport 1994], a constraint is a pair consisting of a *position* and a *translation* in $\mathbb{R}^3$. The latter determines the transformation that should be

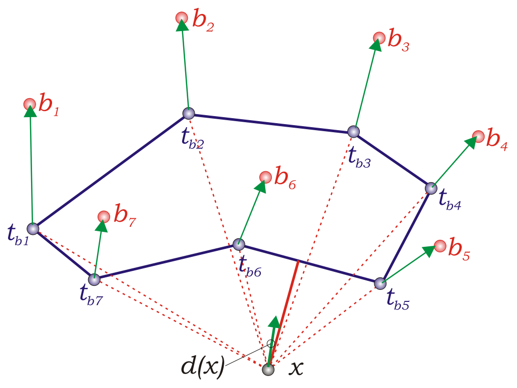


Figure 9.3: *The influence of the handle vertices' transformations ($\mathbf{b}_i - \mathbf{t}_{b\,i}$) on the transformation $\mathbf{d}(\mathbf{x})$ of $\mathbf{x}$ depends on the Hausdorff distance to the generalised handle (thick red line) rather than on the distance of $\mathbf{x}$ to each handle vertex (thin, dashed red line).*

propagated into the region of influence, whereas the former determines the parametrisation of the environment (over which the shape functions are defined) via its distance field.

Instead, we define a generalised constraint $\mathcal{C}$ to be an n-tuple of a subset of $\mathcal{B}$, connected by n line-segments L, together with their respective translations $\mathbf{b} - \mathbf{t}_b$ (see fig. 9.3). Just as the point constraints from [Borrel & Rappoport 1994], this generalised handle defines for each point $\mathbf{x} \in \mathcal{P}$ a translation $\mathbf{d}_h(\mathbf{x})$ and a corresponding weight $\alpha(\mathbf{x})$.

To incorporate this handle type, we need to define a different propagation of the handle vertices' displacement to the region of influence. This also necessitates a new definition of the object-inherent parametrisation of the shape function. Instead of parameterising the object via the distance field with respect to the constraints' vertices, we use the well-known Hausdorff distance $D(x, L)$ of a point x to a set of lines L, defined as the distance of the point to the closest point on L. Thus the influence $\alpha(\mathbf{x})$ at a point $\mathbf{x}$ is:

$$\alpha(\mathbf{x}) = \alpha\left(D(\mathbf{x}, L)\right). \tag{9.2}$$

The translation $\mathbf{d}_h(\mathbf{x})$ on the other hand, is interpolated from the elementary constraints $\mathbf{b} - \mathbf{t}_b$. This interpolation needs to preserve – as boundary condition – the displacement of each boundary vertex. Therefore, we use a radial basis function (RBF) interpolation scheme as in [Botsch & Kobbelt 2005], leading to

$$\mathbf{d}_h(\mathbf{x}) = \sum_{i=1}^{K} \varphi(\gamma(\mathbf{t}_{b\,i}, \mathbf{x}))\mathbf{d}_i, \tag{9.3}$$

with the Gaussian RBF

$$\varphi(t) = e^{-\left(\frac{t}{t_{max}}\right)^2}, \tag{9.4}$$

and the boundary constraints $\mathbf{d}(\mathbf{t}_{b\,i}) = \mathbf{b}_i - \mathbf{t}_{b\,i}$, where t_{max} is set to the length of the template's bounding box diagonal and $\gamma(\mathbf{p}, \mathbf{q})$ is the distance between $\mathbf{p}$ and $\mathbf{q}$ (see below). To satisfy the boundary constraints the partial displacements are calculated via computation of the matrix inverse of

$$\mathbf{A} = \Big(\varphi(\gamma(\mathbf{t}_{b\,i}, \mathbf{t}_{b\,j}))\Big)_{\substack{i=1,\dots,K \\ j=1,\dots,K}}. \tag{9.5}$$

For many close-by handles this inversion is more stable than that in chapter 8 since the matrix depends on the Gaussian RBF – which quickly falls off with increasing distance. Combining the radial basis function interpolation with a user specified shape function yields a robust but nevertheless highly flexible editing metaphor.

9.5 The Modelling Framework

With an appropriate surface prior in place, the modelling phase enables the user to sketch the coarse geometric layout of the surface in the region to be reconstructed. In contrast to other shape modelling applications, the requirements for a modelling method are here somewhat different and features such as volume preservation, detail preservation, energy minimisation etc. are neither desired nor necessary. Since the resulting surface is only guiding the surface reconstruction in the automatic completion process, the prime requirements for our modelling method are intuitiveness, interactivity and flexibility.

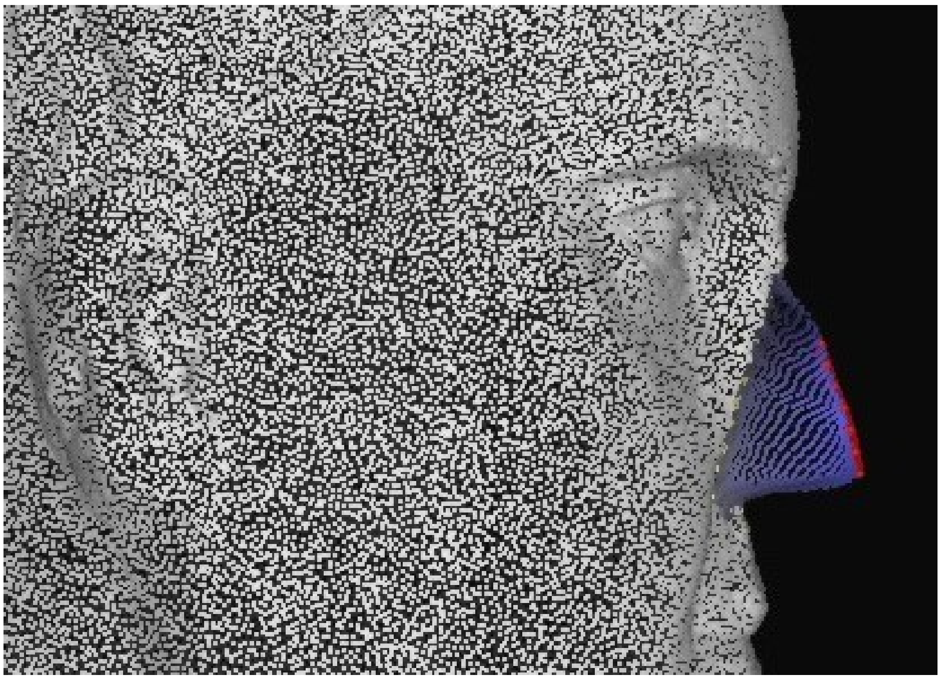


Figure 9.4: *Illustration of a typical editing operation using a generalised handle.*

As stated above, the new modelling phase adopts for *flexibility* the free-form modelling approach presented in the chapter 8, that features interactive modulation of the region of influence and the shape of the edit using shape functions. Targeting at improved *simplicity* and ease of use and drawing upon ideas from [Nealen et al. 2005], this method includes the generalised handles from the previous section, such that feature-line edits as illustrated in figure 9.4 are easily and very efficiently performed.

To this end, the new modelling metaphor will feature two editing modes, both of which employ the generalised constraints from the previous section as *handles.*

With **rigid handles**, the user first selects a handle on the object, either by picking a single point, by drawing a line on the object or by selecting a whole region of the object. Subsequently, a translation and rotation is prescribed for the handle by picking and dragging any point on the handle, thereby defining a transformation matrix. With the distance field from the previous section

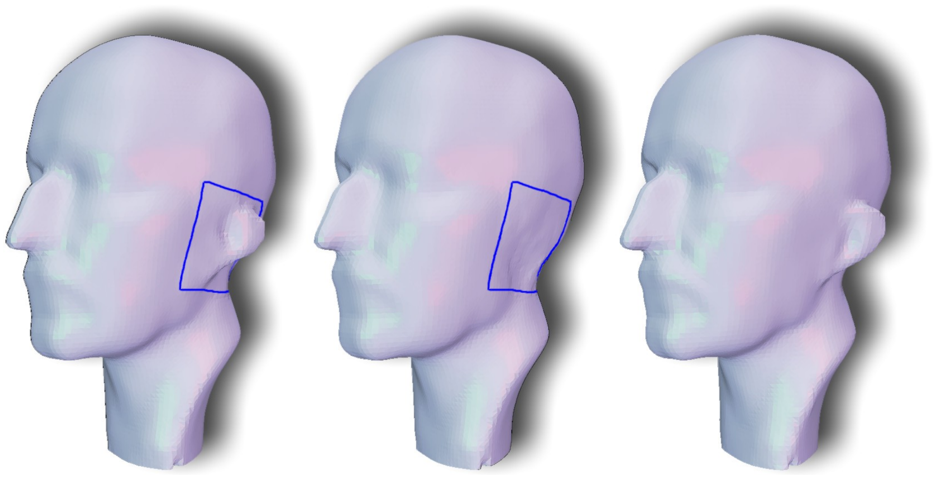


Figure 9.5: *Max Planck-Model. Left: Reconstructed with our approach. Middle: Reconstructed with automatic completion without guidance modelling. The blue line indicates the region that was replaced by a generic template. Right: Original Max Planck-Model. (All reconstructed from the vertices only using MLS and standard Marching Cubes.)*

and the freely definable shape function α, the transformation for a point $\mathbf{x}$ is finally computed as a simple blending

$$\mathbf{x}_{new} = \alpha(\mathbf{x})\mathbf{T}\mathbf{x} \; + \; (1 - \alpha(\mathbf{x}))\mathbf{x},$$

delivering a very effective method for large and fine scale edits.

Sometimes, however, changing the form of the handle itself is more effective than consecutive edits with a rigid handle. Therefore, with **non-rigid handles** the user can also define separate translations for each of the handle vertices. This way, fine-tuned edits (e.g. such as to determine how far the nose of the MaleHead in figure 9.4 is "hooked") can be performed. These translations are then propagated to the point set as described in section 9.4.

In both editing modes, however, the region of influence of the editing operation has to be constrained to the inserted template, with smooth transition to the non-transformed part of the object. To fix the boundary at its position, we redefine the distance field, via which the influence is parameterised, and also incorporate the distance to the boundary. Thus we have:

$$\alpha(\mathbf{x}) = \alpha\left(\frac{D(\mathbf{x}, H)}{D(\mathbf{x}, H) + D(\mathbf{x}, L)}\right), \tag{9.6}$$

where $D(\mathbf{x}, H)$ is the Hausdorff distance of a point $\mathbf{x}$ to the handle H and L is the boundary of the template.

9.5.1 Geodesic Distances for Point Clouds

In the given application setting, we are primarily interested in the modification and restoration of objects represented as unstructured point clouds. Unfortunately, defining geodesic distances on point sets is problematic, and yet the claim that this type of parameterisation is generally more appropriate for shape function modelling than euclidian distances still holds. Fortunately, a strict and precise computation of geodesic distances is not required for this purpose (as was already exploited in chapter 8), and an approximation will suffice.

To this end, a proximity graph based on each point's k nearest neighbours is constructed as suggested by [Klein & Zachmann 2004]: Let $\mathbf{p}$ be a point in the point set and $\mathcal{N}_k(\mathbf{p})$ be the set of the k nearest neighbours of $\mathbf{p}$, then this graph contains an edge $(\mathbf{p}, \mathbf{q})$ iff $\mathbf{q}$ is one of the k nearest neighbours of $\mathbf{p}$. As explained in section 6.3, it is beneficial to symmetrise this graph (and the neighbourhoods correspondingly), in particular for point clouds with varying sampling density.

The "**geodesic**" **distance** between two points of the point cloud is then computed as the length of the shortest path along this proximity graph. Please note that for parameterising the shape function, all points in the ROI can efficiently be computed using a simple breadth first search. It is also worth noting that the error induced by evaluating the distance *along* the graph's edges can be expected to be very small, as (unlike in the triangle mesh case) the vertices in the graph constitute a dense sampling of the underlying surface. Nevertheless, this is only an approximation of the geodesic distance on the approximated surface. However, it still delivers a parameterisation far superior for most modelling situations to those based on Euclidean distances.

9.5.2 Dynamic Point Insertion

It was argued in section 8.5, that editing operations typically necessitate the refinement of triangle meshes in the edited regions. In principle, the same statement also holds for the editing of point sets. In addition, the distance between neighbouring points may have increased significantly after an editing operation. Therefore, the neighbourhood graph is traversed and for each triple of neighbouring points (where each point is a neighbour of the two other), additional points are generated to preserve a minimal neighbour distance δ_{min}. Since another neighbour point may already fill the space inside this (virtual) triangle, additional points are only inserted if their distance is larger than $\frac{\delta_{min}}{2}$ from existing points. After this point insertion, the neighbourhood graph is rebuilt and new normals are estimated for both, the inserted and the original, edited points. Note that it would also be possible to use more sophisticated resampling strategies, e.g. the dynamic resampling method presented in [Pauly et al. 2003], but since the template is only used as guidance surface for the completion step, a more or less regular point distribution is necessary only for modelling convenience.

9.6 Surface Completion

With a suitable guidance surface at hand, we are now able to turn our defective model to an automatic surface completion algorithm and let this automatism reconstruct the missing surface. To this end, we use fragment-based surface completion.

In chapter 7, so-called two-layer descriptors (cf. figure 7.10) are used to identify and compare geometric properties of the surface fragments. In these, the top layer constitutes a local regular resampling of a fragment, which itself is a subset of the point set on level l in the scale-space representation of the given object.

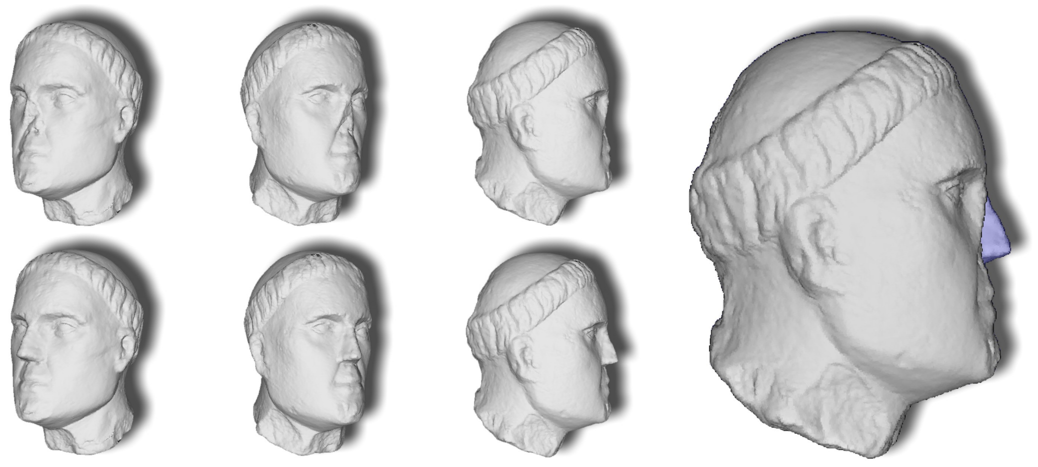


Figure 9.6: *MaleHead data set: The original model (top row) and the reconstruction result (bottom row); the rightmost image emphasises the recreated nose feature (image overlay).*

At the same time, the bottom layer encodes the local geometric properties of the point set on the next coarser level $l + 1$. By formulating the surface completion as an hierarchical algorithm, completing the surface on coarse levels first, and consecutively on the finer and finer levels, the previous completions can be exploited and the corresponding information can be transferred to the next finer levels.

The fundamental weakness of such an iterative process naturally is the starting point of the algorithm, namely the coarsest level. There, one cannot rely on any pre-completed surface to draw information about the geometric layout from. Instead, the required guidance is automatically computed using an extended moving least squares (MLS [Levin 1998]) approach, that was enhanced to prevent the otherwise undesired behaviour of the MLS-surface in the vicinity of insufficient sampling. This way, reasonable, yet only smooth results can be achieved on coarse levels, where the surrounding of the hole is comparatively flat. As demonstrated in the David Head example (see figures 9.8 and 7.15), these smooth guidance surfaces are often not expressive enough to suggest to the next finer level's completion the existence or propaga-

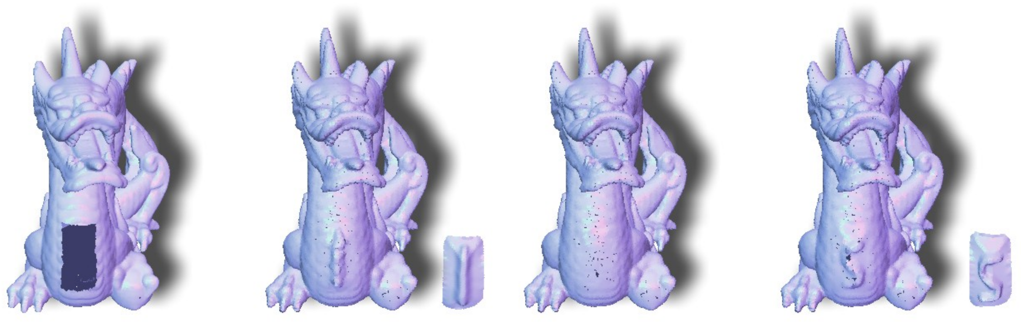


Figure 9.7: *Reconstruction result of the Dragon data set. From left to right: A hole is cut into the dragon's back, the coarsest level with a modelled template prior, the reconstruction using this template, and, for comparison, the original model.*

tion of a feature. In addition to that, a smooth guidance surface in some cases makes it even impossible to detect real symmetry existing in the model, as can be seen in the Max Planck-model (figure 9.5). This is due to the fact that the coarsest level approximation of the surface feature *existent* in the object (the example) differs drastically from the inserted smooth surface patch in the missing similar region of the object.

In contrast, the surface template as modelled by the user is capable of inserting those pieces of information into the algorithm. Therefore, the modelled surface template is incorporated into the coarsest level's guidance surface. Please note that in case of real symmetry, obviously the surface patch still is not *exactly* matching the coarse approximation of the corresponding existent feature. Demanding this would require the user to be unrealistically precise during the modelling phase and would lead our whole algorithm setting ad absurdum.

Since the modified MLS approximation can be expected to have lower confidence on coarsest level than the approximations from higher levels, the weights of the bottom layer in the two-layer-descriptor were increased during the hierarchical completion process. Instead, here we keep the weight of the guidance surface constant to make sure that the users' modelling input is adequately accounted for. Considering the limited accuracy of the modelling

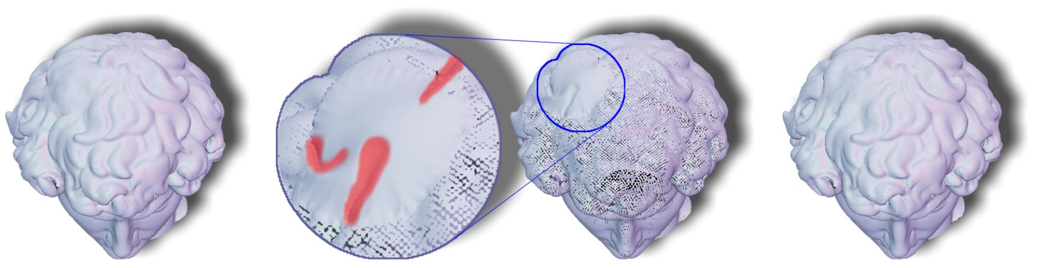


Figure 9.8: *Reconstructed David Head-model. Here, a large piece of david's hair was artificially removed. Without user-interaction, the automatic completion fails to propagate the prominent features from the context into the hole region (left), whereas with only a few very coarse sketches (middle) these structures are adequately reconstructed.*

operation itself, however, and to emphasise its sketch-like character we reduce the descriptor resolution for the bottom layer to half the resolution of the top layer.

This way, even coarse sketches can be used to indicate the presence of features that would otherwise be "overseen" by the completion automatism. Please note also that during the completion phase, all inserted points are attributed with confidence values less than one, such that interested users can always distinguish inserted from "original" parts of the object.

9.7 Results

To evaluate the presented method, it was compared to surface inpainting without user-generated guidance template using various data sets of point sampled geometry. Some of the objects used for evaluation exhibit comparably large defective regions in semantically important parts of the model. Reconstructing the surface after removing the defective region using automatic hole filling algorithms without additional semantic knowledge results in missing features, even though the fine scale detail is preserved. This is especially visible for the Max Planck-Model where the left ear was removed (figure 9.5) and the MaleHead data set (figure 9.6), where the nose broke off and was reconstructed. In both examples the missing surface region cannot be reconstructed without a user-generated guidance surface. In case of the Max Planck-Model, this is due to the fact that the required symmetry relation exists on a scale too large to be captured in the scale space representation of the model. Therefore the hole is patched smoothly, whereas already a coarse sketch (using only two atomic editing operations) to indicate the location of the second ear is sufficient to achieve the desired result. For the MaleHead data set, the problem is slightly different, since the missing nose is a singular feature not to be found anywhere else in the object. Nevertheless, the surface completion algorithm recreates nicely the textural properties of the reconstructed nose on the basis of the more or less accurately modelled template.

In the last two examples of this chapter, the surrounding area does contain enough information for the surface completion to produce reasonable results, even without an expressive template prior. However, the fully automatic reconstruction cannot determine how to propagate the prominent features at the boundary into the hole region. This knowledge can be incorporated into the repairing process with very rough sketches to continue the most important feature lines, as shown in figure 9.8, where the course of the cavities in the hair of the David model is sketched and the resulting repair outperforms that without user guidance depicted in 7.15. Figure 9.7, on the other hand, demonstrates the creative aspect of our approach. Here, the thorn ridge from the dragons back is indicated roughly and automatically elaborated on the front.

Conclusions

The development of 3D data acquisition and the use of digital object representations offer distinctive opportunities for the conservation, research, and dissemination of our cultural heritage. It is therefore not surprising that the projects digitally processing objects of cultural heritage are manifold. Still, most of these projects are implemented with considerable research from computer science groups worldwide; such projects implemented autonomously by the cultural heritage community are rare.

Before this background, this thesis presented methods for geometry processing that aim at bridging the gap between technology that is powerful, yet hard to use in practice, and the needs (and proficiency) prevailing in research and application fields different from computer science. The basic goal of the research presented here was to come up with solutions along the data acquisition pipeline that

- are fully automatic wherever possible and
- provide intuitive means for user interaction everywhere else.

In this respect, in particular the fully automatic registration approach from chapter 4, and the methods for 3D data acquisition via dense sets of photographic images presented in chapter 5 of this thesis can be considered a success. Both are completely interaction free and allow – aside from degenerate and pathologic cases – an even supervision-free reconstruction of the acquired geometry. The most distinctive advantage of the feature-based registration in chapter 4 is that it exploits in addition to the acquired geometry also synchronously recorded photographic images of the object parts represented in the single range images. This approach offers various advantages over pure geometry-based techniques. On the one hand, it benefits from the very successful and robust feature detection techniques available today for images to identify correspondences even in cases of little overlap, and / or considerable changes in perspective and scaling. This way the presented method can even register geometry-featureless object parts, such as parts of rotationally or otherwise symmetric objects. On the other hand, the registration can be performed very efficiently, as it offers various stages of (pre-)alignment with different domains of the optimisation functions, taking into account various subsets of the set of all points to be registered, in particular so-called *feature surface elements*. This enables also a relaxation method that allows for a synchronous registration of all range images.

Inspired from exemplar-based techniques in 2D image processing, chapter 7 introduced a novel method for filling holes in structured point set surfaces. In order to be able to recognise and exploit similarity and coherence properties in the object, we derived target and candidate fragments, each living in their specific scale with a naturally defined fragment size that is well correlated to the respective scale of the filling operations. In addition to that, the fragments are defined in local frames, thereby making our algorithm insensitive to similarity transformations as rotation, translation and scaling. As a consequence of the hierarchical formulation based on a scale-space representation of the object, the completion algorithm is able to robustly identify and exploit similarity relations between the region of interest and possibly various other locations on the surface, depending on the respective scale.

The last part of this thesis focussed on the interactive deformation of the 3D models. The idea behind this was not only to derive a modelling framework for creative applications like animation but also to enable a novel way to let a user's expertise influence and steer the automatic completion from chapter 7. This resulted in a powerful combination of intuitive, only sketch-like editing with an automatic completion to derive plausible completed models according to the user's expertise (or imagination).

In order to assess the aptness, quality or performance of a modelling paradigm, the following criteria are typically called upon:

- High quality

- Precise Control
- Locality
- High Flexibility
- Intuitiveness / Predictability

While the last four criteria are hardly arguable, notions do differ what is to be understood as *high quality* editing. This diffuse requirement, often also termed (equally diffuse) *Fair* Surface Design, varies with the scientific context and application. Some require smoothness to a certain degree in the differential geometry sense, some require minimising properties for certain energy functionals like thin plate energy or Willmore flow. If nothing else is specified, a fair surface is typically understood to be one that corresponds to the simplest gestalt principle [Sapidis 1994]. The modelling approach introduced in chapter 8 leaves the smoothness of the editing operation completely at the user's discretion. This is realised by including interactively and freely adjustable *shape functions*.

The *guidance modelling* presented in chapter 9 resembles to a certain extent the popular multi-resolution modelling approach presented in various papers, such as [Kobbelt et al. 1998; Lee 1999], among others. In these approaches, deformations are also performed on coarse levels, defining the large scale layout of the new shape, whereas the fine details are preserved. However, the main difference is that with these approaches only details that are existent in the modelled area in the first place can be preserved, while the new approach synthesises these details based on an analysis of the context of the hole *and* on the modelled shape prior.

This approach here also relates to so-called *Surface Coating* [Sorkine et al. 2004], which transfers detail coefficients from a *source* to a *target* region. This coating, however, requires that the underlying surface is fully modelled, and therefore is comparable only for singular features, whereas the present method handles these cases satisfyingly and is, in addition to that, capable of identifying and exploiting similarity and coherence properties of the object.

The methods presented in this thesis follow the basic layout of a 3D geometry acquisition and exploitation pipeline and do build upon one another in this respect. Nevertheless, each of them can be used as a stand-alone solution to the specific problem setting. Moreover, in case a particular application setting requires a different treatment for one of the partial problems along the acquisition pipeline, any of the presented algorithms can of course be modified, extended, and even replaced by a more custom-tailored or appropriate solution, if available, without affecting the other stages. In this sense, the proposed approaches are completely modular.

Bibliography

Adamson, A., & Alexa, M. 2003. Approximating and intersecting surfaces from points. In *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, Eurographics Association, 230–239. 19

Adamson, A., & Alexa, M. 2004. Approximating bounded, non-orientable surfaces from points. In *Proceedings of the Shape Modeling International 2004 (SMI'04)*, IEEE Computer Society, 243–252. 19

Akca, D. 2003. Full automatic registration of laser scanner point clouds. In *Optical 3-D Measurement Techniques VI*, A. Gruen, & H. Kahmen, Eds., vol. 1, 330–337. 30

Alexa, M., & Adamson, A. 2004. On normals and projection operators for surfaces defined by point sets. In *Proceedings of Eurographics Symposium on Point-Based Graphics*, 150–155. 19

Allen, B., Curless, B., & Popovic, Z. 2003. The space of human body shapes: reconstruction and parameterization from range scans. *ACM Trans. Graph. 22*, 3, 587–594. 162

Amenta, N., Bern, M., & Eppstein, D. 1998. The crust and the -skeleton: Combinatorial curve reconstruction. *Graphical models and image processing: GMIP 60*, 2, 125–135. 73, 75

Angelidis, A., & Cani, M.-P. 2002. Adaptive implicit modeling using subdivision curves and surfaces as skeletons. In *Solid Modelling and Applications*, ACM. Saarbrucken, Germany. 129

Angelidis, A., Jepp, P., & Cani, M.-P. 2002. Implicit modeling with skeleton curves: Controlled blending in contact situations. In *Shape Modeling International*, IEEE Computer Society Press, ACM. Banff, Alberta, Cananda. 129

ASHBROOK, A. P., & FISHER, R. B. 1997. Constructing models of articulating objects: range data partitioning. In *Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, IEEE Computer Society, 164. 33

BALLESTER, C., BERTALMIO, M., CASELLES, V., SAPIRO, G., & VERDERA, J. 2001. Filling-in by joint interpolation of vector fields and gray levels. *IEEE Transactions on Image Processing 10*, 8 (August), 1200–1211. 98

BARR, A. H. 1984. Global and local deformations of solid primitives. In *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 21–30. 125

BAUMGART, B. G., 1974. Geometric modeling for computer vision. Technical Report AIM-249, Artificial Intelligence Lab, Stanford University, October. 60

BERGEVIN, R., SOUCY, M., GAGNON, H., & LAURENDEAU, D. 1996. Towards a general multi-view registration technique. *IEEE Trans. Pattern Anal. Mach. Intell. 18*, 5, 540–547. 35

BESL, P. J., & MCKAY, N. D. 1992. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 2, 239–256. 16, 32, 35, 162

BLAIS, G., & MARTIN, L. 1994. Registering multiview range data to create 3D computer objects. Tech. Rep. CIM-93-16, McGill Centre for Intelligent Machines, Mar. 35

BLINN, J. F. 1982. A generalization of algebraic surface drawing. *ACM Transactions on Graphics (TOG) 1*, 3, 235–256. 129

BLOOMENTHAL, J., & SHOEMAKE, K. 1991. Convolution surfaces. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, ACM Press, 251–256. 129

Bloomenthal, J., & Wyvill, B. 1990. Interactive techniques for implicit modeling. In *SI3D '90: Proceedings of the 1990 symposium on Interactive 3D graphics*, ACM Press, New York, NY, USA, 109–116. 129

Bloomenthal, J. 1988. Polygonization of implicit surfaces. *Comput. Aided Geom. Des. 5*, 4, 341–355. 24

Bloomenthal, J. 1997. *Introduction to Implicit Surfaces.* Morgan Kaufmann Publishers, Inc. 11, 129, 140

Bonet, J. S. D. 1997. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Computer Graphics*, ACM SIGGRAPH, 361–368. 97

Borrel, P., & Bechmann, D. 1991. Deformation of n-dimensional objects. In *Proceedings of the first ACM symposium on Solid modeling foundations and CAD/CAM applications*, ACM Press, 351–369. 129

Borrel, P., & Rappoport, A. 1994. Simple constrained deformations for geometric modeling and interactive design. *ACM Transactions on Graphics (TOG) 13*, 2, 137–155. 129, 132, 133, 162, 163

Botsch, M., & Kobbelt, L. P. 2004. An intuitive framework for real-time freeform modeling. *ACM Trans. Graph. 23*, 3, 630–634. 127

Botsch, M., & Kobbelt, L. P. 2005. Real-time shape editing using radial basis functions. *Computer Graphics Forum 24*, 3, 611 – 621. 22, 73, 127, 164

Botsch, M., Pauly, M., Gross, M., & Kobbelt, L. 2006. Primo: Coupled prisms for intuitive surface modelling. In *Eurographics Symposium on Geometry Processing (TO APPEAR).* 127, 150

Buhmann, M. D. 2003. *Radial Basis Functions – Theory and Implementations.* No. 12 in Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press. (ISBN-13: 9780521633383 — ISBN-10: 0521633389). 22

B.WYVILL, C.MCPHEETERS, & G.WYVILL. 1986. Data structure for soft objects. *The Visual Computer 2*, 4, 227–234. 129

CALLIERI, M., CIGNONI, P., GANOVELLI, F., MONTANI, C., PINGI, P., & SCOPIGNO, R. 2003. Vclab's tools for 3d range data processing. In *4th International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage (VAST2003) and First EUROGRAPHICS Workshop on Graphics and Cultural Heritage*. 30

CANI, M.-P., & HORNUS, S. 2001. Subdivision curve primitives: a new solution for interactive implicit modeling. In *Shape Modelling International*. 129

CANI-GASCUEL, M.-P., & DESBRUN, M. 1997. Animation of deformable models using implicit surfaces. *IEEE Transactions on Visualization and Computer Graphics 3*, 1, 39–50. 141

CANI, M.-P. 1999. Implicit representations in computer animation: A compared study. In *Proceedings of Implicit Surface '99*. Invited paper. 129

CARR, J. C., BEATSON, R. K., CHERRIE, J. B., MITCHELL, T. J., FRIGHT, W. R., MCCALLUM, B. C., & EVANS, T. R. 2001. Reconstruction and representation of 3d objects with radial basis functions. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 67–76. 17, 20, 21, 22, 100

CARR, J. C., BEATSON, R. K., MCCALLUM, B. C., FRIGHT, W. R., MCLENNAN, T. J., & MITCHELL, T. J. 2003. Smooth surface reconstruction from noisy range data. In *GRAPHITE '03: Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, ACM Press, New York, NY, USA, 119–ff. 22, 100

CATMULL, E., & CLARK, J. H. 1978. Recursively generated b-spline surfaces on arbitrary topological meshes. *Computer-Aided Design 10* (November), 350–360. 125

CHAIKIN, G. 1974. Short note: An algorithm for high-speed curve generation. *Computer Graphics and Image Processing 3*, 346–349. 125

CHEN, Y., & MEDIONI, G. 1992. Object modelling by registration of multiple range images. *Image Vision Comput. 10*, 3, 145–155. 35

CHEN, C.-S., HUNG, Y.-P., & CHENG, J.-B. 1999. Ransac-based darces: A new approach to fast automatic registration of partially overlapping range images. *IEEE Trans. Pattern Anal. Mach. Intell. 21*, 11, 1229–1234. 33

CLARENZ, U., DIEWALD, U., & RUMPF, M. 2000. Anisotropic geometric diffusion in surface processing. In *Proceedings of the conference on Visualization '00*, IEEE Computer Society Press, 397–405. 113

CLARENZ, U., DIEWALD, U., DZIUK, G., RUMPF, M., & RUSU, R. 2004. A finite element method for surface restoration with smooth boundary conditions. *Computer Aided Geometric Design 21*, 5, 427–445. 100

COOMBE, G., HANTAK, C., LASTRA, A., & GRZESZCZUK, R. 2005. Online reconstruction of surface light fields. In *Eurographics Symposium on Rendering*. 56

COQUILLART, S. 1990. Extended free-form deformation: a sculpturing tool for 3d geometric modeling. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, ACM Press, 187–196. 125

COSTELLO, R. B., Ed. 1991. *Random House Webster's College Dictionary*. Random House. 7

CRIMINISI, A., PÉREZ, P., & TOYAMA, K. 2003. Object removal by exemplar-based inpainting. In *Conference on Computer*

Vision and Pattern Recognition (CVPR 2003), IEEE Computer Society, Madison, WI, USA, 721–728. 98, 109

Cunnington, S., & Stoddart, A. 1999. N-view point set registration: A comparison. In *British Machine Vision Conference*, vol. 1, 234–244. 35

Curless, B., & Levoy, M. 1996. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM Press, 303–312. 100

Dana, K. J., Nayar, S. K., Ginneken, B. V., & Koenderink, J. J. 1997. Reflectance and texture of real-world surfaces. In *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, IEEE Computer Society, Washington, DC, USA, 151–157. 53

Davis, J., Marschner, S. R., Garr, M., & Levoy, M. 2002. Filling holes in complex surfaces using volumetric diffusion. In *Proceedings of the 1st Int. Symp. on 3D Data Processing Visualization and Transmission*, IEEE Computer Society, Padova, Italy, G. M. Cortelazzo, & C. Guerra, Eds., 428–438. 100

Debevec, P., Hawkins, T., Tchou, C., Duiker, H.-P., Sarokin, W., & Sagar, M. 2000. Acquiring the reflectance field of a human face. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 145–156. 56

Degener, P., Meseth, J., & Klein, R. 2003. An adaptable surface parametrization method. In *The 12th International Meshing Roundtable 2003*. 64, 67

DePiero, F. W. 2003. Deterministic surface registration at 10hz based on landmark graphs with prediction. In *14th British Machine Vision Conf. (BMVC2003)*. 34

Desbrun, M., Meyer, M., Schröder, P., & Barr, A. H. 1999. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 317–324. 113

Dey, T. K., & Giesen, J. 2001. Detecting undersampling in surface reconstruction. In *Proceedings of the seventeenth annual symposium on Computational geometry*, ACM Press, 257–263. 75, 92

Drori, I., Cohen-Or, D., & Yeshurun, H. 2003. Fragment-based image completion. *ACM Trans. Graph. 22*, 3, 303–312. 98, 110

Duchon, J. 1977. Splines minimizing rotation-invariant semi-norms in sobolev spaces, constructive theory of functions of several variables. *Lecture Notes in Mathematics, Springer-Verlag, Berlin, 571*, 85–100. 21, 22

Efros, A. A., & Leung, T. K. 1999. Texture synthesis by non-parametric sampling. In *ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2*, IEEE Computer Society, 1033. 97

Eggert, D. W., Fitzgibbon, A. W., & Fisher, R. B. 1998. Simultaneous registration of multiple range views for use in reverse engineering of cad models. *Comput. Vis. Image Underst. 69*, 3, 253–272. 35

Faloutsos, P., van de Panne, M., & Terzopoulos, D. 1997. Dynamic free-form deformations for animation synthesis. *IEEE Transactions on Visualization and Computer Graphics 3*, 3 (/), 201–214. 125

Farin, G. 1990. *Curves and surfaces for computer aided geometric design.* Academic Press Professional, Inc., San Diego, CA, USA. 10, 124

FAUGERAS, O., & HEBERT, M. 1986. The representation, recognition, and locating of 3-d objects. *Int. J. Rob. Res. 5*, 3, 27–52. 33

FELDMAR, J., & AYACHE, N. 1996. Rigid, affine and locally affine registration of free-form surfaces. *Int. J. Comput. Vision 18*, 2, 99–119. 33

FISCHLER, M. A., & BOLLES, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *Commun. ACM*, ACM Press, New York, NY, USA, vol. 24, 381–395. 39

FLEISHMAN, S., COHEN-OR, D., & SILVA, C. T. 2005. Robust moving least-squares fitting with sharp features. *ACM Trans. Graph. 24*, 3, 544–552. 19

FLOATER, M. S., & HORMANN, K. 2005. Surface parameterization: a tutorial and survey. In *Advances in multiresolution for geometric modelling*, N. A. Dodgson, M. S. Floater,, & M. A. Sabin, Eds. Springer Verlag, 157–186. 64

FRISCH, N., & ERTL, T. 2002. Deformation of finite element meshes using directly manipulated free-form deformation. In *Proceedings of the seventh ACM symposium on Solid modeling and applications*, ACM Press, 249–256. 125

FRISKEN, S. F., PERRY, R. N., ROCKWOOD, A. P., & JONES, T. R. 2000. Adaptively sampled distance fields: a general representation of shape for computer graphics. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 249–254. 145

FURUKAWA, R., KAWASAKI, H., IKEUCHI, K., & SAKAUCHI, M. 2002. Appearance based object modeling using texture database: acquisition, compression and rendering. In *EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 257–266. 56

Furukawa, Y., Lazebnik, S., & Ponce, J. 2005. Carved visual hulls for high-accuracy image-based modeling. In *Siggraph Sketches*. 70

Gain, J. E., & Dodgson, N. A. 1999. Adaptive refinement and decimation under free-form deformation. In *Eurographics UK '99*. 147

Gascuel, M.-P. 1993. An implicit formulation for precise contact modeling between flexible solids. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 313–320. 141

Gelfand, N., Mitra, N. J., Guibas, L. J., & Pottmann, H. 2005. Robust global registration. In *Third Eurographics Symposium on Geometry Processing*, 197–206. 33

Gibson, S. F. F. 1998. Constrained elastic surface nets: Generating smooth surfaces from binary segmented data. In *MICCAI '98: Proceedings of the First International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer-Verlag, London, UK, 888–898. 24

Gortler, S. J., Grzeszczuk, R., Szeliski, R., & Cohen, M. F. 1996. The lumigraph. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 43–54. 55

Grauman, K., Shakhnarovich, G., & Darrell, T. 2003. A bayesian approach to image-based visual hull reconstruction. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 187–194. 70

Greissmair, J., & Purgathofer, W. 1989. Deformation of solids with trivariate b-splines. In *Computer Graphics Forum*, 137–148. 147

Gress, A., & Klein, R. 2003. Efficient representation and extraction of 2-manifold isosurfaces using kd-trees. In *The 11th*

Pacific Conference on Computer Graphics and Applications (PG 2003), IEEE Computer Society, 364–376. 24

GRESS, A., & KLEIN, R. 2004. Efficient representation and extraction of 2-manifold isosurfaces using kd-trees. *Graphical Models 66*, 6 (November), 370–397. 24

GUMHOLD, S., WANG, X., & MCLEOD, R. 2001. Feature extraction from point clouds. In *Proceedings of 10th International Meshing Roundtable*, 293–305. 74, 75, 81, 87

GUSKOV, I., SWELDENS, W., & SCHRÖDER, P. 1999. Multiresolution signal processing for meshes. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 325–334. 101, 127

HARRIS, C., & STEPHENS, M. 1988. A combined corner and edge detector. In *Proceedings of The Fourth Alvey Vision Conference, Manchester*, 147–151. 34

HAVEMANN, S. 2005. *Generative Mesh Modeling*. PhD thesis, Technische Universität Braunschweig. 8

HAWKINS, T., COHEN, J., & DEBEVEC, P. 2001. A photometric approach to digitizing cultural artifacts. In *VAST '01: Proceedings of the 2001 conference on Virtual reality, archeology, and cultural heritage*, ACM Press, New York, NY, USA, 333–342. 12, 14, 56

HAWKINS, T., EINARSSON, P., & DEBEVEC, P. 2005. A dual light stage. In *Eurographics Symposium on Rendering*. 14, 56

HEEGER, D. J., & BERGEN, J. R. 1995. Pyramid-based texture analysis/synthesis. In *SIGGRAPH*, 229–238. 97

HIGUCHI, K., HEBERT, M., & IKEUCHI, K. 2001. *Modelling from reality*. Kluwer Academic Publishers, Norwell, MA, USA, ch. Building 3-D models from unregistered range images, 41–75. 33

HOFF, K. E., KEYSER, J., LIN, M., MANOCHA, D., & CULVER, T. 1999. Fast computation of generalized voronoi diagrams using graphics hardware. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 277–286. 106

HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., & STUETZLE, W. 1992. Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, ACM Press, 71–78. 17, 85, 86, 104

HORN, B. 1987. Closed form solution of absolute orientation using unit quaternions. *Journal of the Optical Society A 4*, 4 (April), 629–642. 42

HSU, W. M., HUGHES, J. F., & KAUFMAN, H. 1992. Direct manipulation of free-form deformations. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 177–184. 125

HU, S.-M., ZHANG, H., TAI, C.-L., & SUN, J.-G. 2001. Direct manipulation of ffd: efficient explicit solutions and decomposible multiple point constraints. *The Visual Computer 17*, 6, 370–379. 125

IGARASHI, T., MATSUOKA, S., & TANAKA, H. 1999. Teddy: a sketching interface for 3d freeform design. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 409–416. 129

ISIDORO, J., & SCLAROFF, S. 2003. Stochastic refinement of the visual hull to satisfy photometric and silhouette consistency constraints. In *Proc. 9th International Conference of Computer Vision*, 1335–1342. 70

JIA, J., & TANG, C.-K. 2003. Image repairing: Robust image synthesis by adaptive nd tensor voting. In *Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, IEEE Computer Society, Madison, WI, USA, 643–650. 98

JOHNSON, A. E., & HEBERT, M. 1997. Surface registration by matching oriented points. In *Proceedings of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, IEEE Computer Society, 121. 33

JU, T., LOSASSO, F., SCHAEFER, S., & WARREN, J. 2002. Dual contouring of hermite data. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 339–346. 24

KAMGAR-PARSI, B., JONES, J. L., & ROSENFELD, A. 1991. Registration of multiple overlapping range images: Scenes without distinctive features. *IEEE Trans. Pattern Anal. Mach. Intell. 13*, 9, 857–871. 33

KARBACHER, S., & HÄUSLER, G. 1998. New approach for the modeling and smoothing of scattered 3d data. In *Proceedings of the Conference on 3D Image Capture and Applications*, SPIE, San Jose, CA, USA, R. N. Ellson, & J. H. Nurre, Eds., vol. 3313 of *SPIE Proceedings*, 168–177. 113

KARPENKO, O., HUGHES, J. F., & RASKAR, R. 2002. Free-form sketching with variational implicit surfaces. *Computer Graphics Forum 21*, 3 (September), 585–585. 129

KLEIN, J., & ZACHMANN, G. 2004. Proximity graphs for defining surfaces over point clouds. In *Eurographics Symposium on Point-Based Grahics (SPBG'04)*, 131–138. 19, 104, 167

KLEIN, R. 2000. 3d mesh-editing. In *Dagstuhl Seminar: Image Synthesis and Interactive 3D Graphics*, Michael Cohen, Heinrich Mueller, Claude Puech, Hans-Peter Seidel, vol. 25. http://cg.cs.uni-bonn.de/docs/publications/2000/dagstuhl-presentation.pdf. 126

KOBBELT, L. P., CAMPAGNA, S., VORSATZ, J., & SEIDEL, H.-P. 1998. Interactive multi-resolution modeling on arbitrary meshes. In *SIGGRAPH 98 Conference Proceedings*, Addison Wesley, M. Cohen, Ed., Annual Conference Series, ACM SIGGRAPH, 105–114. ISBN 0-89791-999-8. 126, 127, 175

KOBBELT, L. P., BAREUTHER, T., & SEIDEL, H.-P. 2000. Multiresolution shape deformations for meshes with dynamic vertex connectivity. *Computer Graphics Forum 19*, 3 (August), 249–260. ISSN 1067-7055. 126

KOBBELT, L. P., BOTSCH, M., SCHWANECKE, U., & SEIDEL, H.-P. 2001. Feature sensitive surface extraction from volume data. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 57–66. 24

KOCH, D. 2006. *Simultane Erfassung der Geometrie und Oberflächenreflexionseigenschaften komplexer Objekte mittels eines hemisphärischen Kamerafeldes.* Master's thesis, Universität Bonn, Germany. 14, 57

KRAEVOY, V., & SHEFFER, A. 2005. Template-based mesh completion. In *Eurographics Symposium on Geometry Processing 2005*, 13–22. 100

KRSEK, P., PAJDLA, T., & HLAVAC, V. 2002. Differential invariants as the base of triangulated surface registration. *Comput. Vis. Image Underst. 87*, 1-3, 27–38. 33

LAFORTUNE, E. P. F., FOO, S.-C., TORRANCE, K. E., & GREENBERG, D. P. 1997. Non-linear approximation of reflectance functions. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 117–126. 55

LAURENTINI, A. 1994. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell. 16*, 2, 150–162. 61

LEE, S. 1999. Interactive multiresolution editing of arbitrary meshes. In *Proc. of Eurographics '99*, P. Brunet, & R. Scopigno, Eds., C–73–C82. 128, 175

LENSCH, H. P. A., KAUTZ, J., GOESELE, M., HEIDRICH, W., & SEIDEL, H.-P. 2003. Image-based reconstruction of spatial appearance and geometric detail. *ACM Trans. Graph. 22*, 2, 234–257. 56

LEVIN, D. 1998. The approximation power of moving least-squares. *Math. Comput. 67*, 224, 1517–1531. 169

LEVIN, D. 2003. *Geometric Modeling for Scientific Visualization*. Springer Verlag, ch. Mesh-independent surface interpolation, 37–49. 17

LEVOY, M., & HANRAHAN, P. 1996. Light field rendering. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 31–42. 55

LI, X., & GUSKOV, I. 2005. Multiscale features for approximate alignment of point-based surfaces. In *Third Eurographics Symposium on Geometry Processing*, 217–226. 33

LI, M., MAGNOR, M., & SEIDEL, H.-P. 2003. Hardware-accelerated visual hull reconstruction and rendering. In *In Proceedings of GI03*. 70

LI, M., MAGNOR, M., & SEIDEL, H.-P. 2003. Improved hardware-accelerated visual hull rendering. In *Vision, Modeling, and Visualization 2003*. 70

LIEPA, P. 2003. Filling holes in meshes. In *SGP'03: Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, Eurographics Association, 200–205. 100

LIN, W.-C., HAYS, J. H., WU, C., KWATRA, V., & LIU, Y. 2004. A Comparison Study of Four Texture Synthesis Algorithms on Regular and Near-regular Textures. Tech. Rep. CMU-RI-TR-04-01, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA. 98

Lindeberg, T. 1993. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: a method for focus-of-attention. *Int. J. Comput. Vision 11*, 3, 283–318. 37

Linsen, L., & Prautzsch, H. 2002. Fan clouds - an alternative to meshes. In *Proceedings Dagstuhl Seminar 02151 on Theoretical Foundations of Computer Vision - Geometry, Morphology and Computational Imaging*, Springer-Verlag Berlin Heidelberg, T. Alano, R. Klette,, & C. Ronse, Eds., [10]. 74, 85

Lipman, Y., Sorkine, O., Cohen-Or, D., Levin, D., Rössl, C., & Seidel, H.-P. 2004. Differential coordinates for interactive mesh editing. In *Proceedings of Shape Modeling International*, IEEE Computer Society Press, 181–190. 127

Lipman, Y., Sorkine, O., Levin, D., & Cohen-Or, D. 2005. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph. 24*, 3, 479–487. 127

Llamas, I., Kim, B., Gargus, J., Rossignac, J., & Shaw, C. D. 2003. Twister: A space-warp operator for the two-handed editing of 3d shapes. *ACM Transactions on Graphics 22*, 3 (July), 663–668. 128

Llamas, I., Powell, A., Rossignac, J., & Shaw, C. D. 2005. Bender: a virtual ribbon for deforming 3d shapes in biomedical and styling applications. In *SPM '05: Proceedings of the 2005 ACM symposium on Solid and physical modeling*, ACM Press, New York, NY, USA, 89–99. 128

Lorensen, W. E., & Cline, H. E. 1987. Marching cubes: A high resolution 3d surface construction algorithm. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 163–169. 23

LOWE, D. G. 1999. Object recognition from local scale-invariant features. In *Proc. of the International Conference on Computer Vision ICCV, Corfu*, 1150–1157. 37, 39

LOWE, D. G. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2 (11), 91–110. 37

LUCAS, B. D., & KANADE, T. 1981. An iterative image registration technique with an application to stereo vision (darpa). In *Proceedings of the 1981 DARPA Image Understanding Workshop*, 121–130. 34

MALZBENDER, T., GELB, D., & WOLTERS, H. 2001. Polynomial texture maps. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 519–528. 14, 52, 57, 62

MARKOSIAN, L., COHEN, J. M., CRULLI, T., & HUGHES, J. 1999. Skin: a constructive approach to modeling free-form shapes. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 393–400. 129

MARTIN, W. N., & AGGARWAL, J. K. 1983. Volumetric description of objects from multiple views. *IEEE Trans. Pattern Analysis and Machine Intelligence 5*, 2, 150–158. 60

MASUDA, T., SAKAUE, K., & YOKOYA, N. 1996. Registration and integration of multiple range images for 3-d model construction. In *Proceedings of the 1996 International Conference on Pattern Recognition (ICPR '96) Volume I*, IEEE Computer Society, 879. 35

MATUSIK, W., BUEHLER, C., RASKAR, R., GORTLER, S. J., & MCMILLAN, L. 2000. Image-based visual hulls. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 369–374. 61

Matusik, W., Pfister, H., Ziegler, R., Ngan, A., & McMillan, L. 2002. Acquisition and rendering of transparent and refractive objects. In *EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 267–278. 56

McAllister, D. K. 2002. *A Generalized Surface Appearance Representation for Computer Graphics.* PhD thesis, Dept. of Computer Science, University of North Carolina at Chapel Hill. 57

Mikolajczyk, K., & Schmid, C. 2003. A performance evaluation of local descriptors. In *Conference on Computer Vision and Pattern Recognition CVPR*, vol. 2, IEEE, 257–263. 37

Moenning, C., & Dodgson, N. 2004. Intrinsic point cloud simplification. In *Proc. 14th GraphiCon*, vol. 14. 74

Moreton, H. P., & Séquin, C. H. 1992. Functional optimization for fair surface design. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 167–176. 127

Müller, G., Meseth, J., & Klein, R. 2004. Fast environmental lighting for local-pca encoded btfs. In *Computer Graphics International 2004 (CGI 2004)*, IEEE Computer Society, 198–205. 63

Müller, G., Meseth, J., Sattler, M., Sarlette, R., & Klein, R. 2005. Acquisition, synthesis and rendering of bidirectional texture functions. *Computer Graphics forum 24*, 1 (March), 83–109. 63

Nealen, A., & Alexa, M. 2003. Hybrid texture synthesis. In *EGRW '03: Proceedings of the 14th Eurographics workshop on Rendering*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 97–105. 97

NEALEN, A., SORKINE, O., ALEXA, M., & COHEN-OR, D. 2005. A sketch-based interface for detail-preserving mesh editing. *ACM Trans. Graph. 24*, 3, 1142–1147. 129, 165

NICOLL, A., MESETH, J., MÜLLER, G., & KLEIN, R. 2005. Fractional fourier texture masks: Guiding near-regular texture synthesis. *Computer Graphics Forum 24*, 3 (September), 569–579. 98

NOVOTNI, M., & KLEIN, R. 2002. Computing geodesic distances on triangular meshes. In *The 10-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*, 341–347. 138, 149

OHTAKE, Y., BELYAEV, A., ALEXA, M., TURK, G., & SEIDEL, H.-P. 2003. Multi-level partition of unity implicits. *ACM Trans. Graph. 22*, 3, 463–470. 17, 22

OPALACH, A., & CANI-GASCUEL, M. 1997. Local deformations for animation of implicit surfaces. In *13th Spring Conference on Computer Graphics*, W. Straßer, Ed., 85–92. 129, 141

PAULY, M., & GROSS, M. 2001. Spectral processing of point-sampled geometry. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 379–386. 101

PAULY, M., KEISER, R., KOBBELT, L. P., & GROSS, M. 2003. Shape modeling with point-sampled geometry. *ACM Trans. Graph. 22*, 3, 641–650. 73, 128, 168

PAULY, M., MITRA, N., GIESEN, J., GROSS, M., & GUIBAS, L. 2005. Example-based 3d scan completion. In *Third Eurographics Symposium on Geometry Processing*, 23–32. 100, 162

PENROSE, R. 1955. A generalized inverse for matrices. In *Proc. Cambridge Philos. Soc.*, 406–413. 133

PERLIN, K. 1985. An image synthesizer. In *Proc. of the 12th annual conference on Computer graphics and interactive techniques*, vol. 19(3), 287–296. 97

Pfister, H., Zwicker, M., van Baar, J., & Gross, M. 2000. Surfels: surface elements as rendering primitives. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 335–342. 9

Pollefeys, M., Gool, L. V., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., & Koch, R. 2004. Visual modeling with a hand-held camera. *International Journal of Computer Vision 59*, 3, 207–232. 56

Portilla, J., & Simoncelli, E. P. 2000. A parametric texture model based on joint statistics of complex wavelet coefficients. *Intern. Journal of Computer Vision 40*, 1, 49–70. 97

Pulli, K. 1997. *Surface reconstruction and display from range and color data*. PhD thesis, University of Washington. Chairperson-Linda G. Shapiro. 56

Pulli, K. 1999. Multiview registration for large data sets. In *Proceedings of the International Conference on 3D Digital Imaging and Modeling*, 160–168. 35

Raffin, Neveu, & Jaar. 2000. Curvilinear displacement of free-form-based deformation. *The Visual Computer 16*, 1, 38–46. 129, 130, 137

Robertson, C., & Fisher, R. B. 2002. Parallel evolutionary registration of range data. *Comput. Vis. Image Underst. 87*, 1-3, 39–50. 33

Rodrigues, M., Fisher, R., & Liu, Y. 2002. Special issue on registration and fusion of range images. *Comput. Vis. Image Underst. 87*, 1/2/3, 1–7. 32

Roth, G. 1999. Registering two overlapping range images. In *Proceedings of the Second International Conference on 3-D Digital Imaging and Modeling*, 191–200. 34

RUSINKIEWICZ, S., & LEVOY, M. 2001. Efficient variants of the ICP algorithm. In *Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling (3DIM-01)*, 145–152. 32

SAPIDIS, N. S. 1994. *Designing Fair Curves and Surfaces: Shape Quality in Geometric Modeling and Computer-Aided Design*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA. 127, 175

SAPPA, A., & GARCÍA, M. 2000. Incremental multiview integration of range images. In *Proceedings of the International Conference on Pattern Recognition*, vol. 1, 546–549. 35

SATTLER, M., SARLETTE, R., & KLEIN, R. 2003. Efficient and realistic visualization of cloth. In *EGRW '03: Proceedings of the 14th Eurographics workshop on Rendering*, Eurographics Association, 167–177. 58

SATTLER, M., SARLETTE, R., , MÜCKEN, T., & KLEIN, R. 2005. Exploitation of human shadow perception for fast shadow rendering. In *APGV 2005: Proceedings of the ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization*, ACM, 131–134. 70

SAVCHENKO, V., & KOJEKINE, N. 2002. An approach to blend surfaces. *Advances in Modeling, Animation and Rendering ISBN: 1852336544*, 139–150. 101

SCHMITT, F., & BENJEMAA, R., 1997. Fast global registration of 3D sampled surfaces using a multi-Z-buffer technique, Sept. 12. 35

SEDERBERG, T. W., & PARRY, S. R. 1986. Free-form deformation of solid geometric models. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, ACM Press, 151–160. 125

SEO, J. K., SHARP, G. C., & LEE, S. W. 2005. Range data registration using photometric features. In *CVPR '05: Proceed-*

ings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2, IEEE Computer Society, Washington, DC, USA, 1140–1145. 50

Sharf, A., Alexa, M., & Cohen-Or, D. 2004. Context-based surface completion. *ACM Trans. Graph. 23*, 3, 878–887. 101

Sharp, G. C., Lee, S. W., & Wehe, D. K. 2002. Icp registration using invariant features. *IEEE Trans. Pattern Anal. Mach. Intell. 24*, 1, 90–102. 33

Shu, R., Zhou, C., & Kankanhalli, M. S. 1995. Adaptive marching cubes. *The Visual Computer 11*, 4, 202–217. 24

Slabaugh, G. G., Culbertson, W. B., Malzbender, T., Stevens, M. R., & Schafer, R. W. 2004. Methods for volumetric reconstruction of visual scenes. *Int. J. Comput. Vision 57*, 3, 179–199. 70

Sorkine, O., Lipman, Y., Cohen-Or, D., Alexa, M., Rössl, C., & Seidel, H.-P. 2004. Laplacian surface editing. In *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, Eurographics Association, 179–188. 127, 175

Sorkine, O. 2005. Laplacian mesh processing. In *Eurographics 2005 - State of the Art Reports*, 53–70. 127

Stein, F., & Medioni, G. 1992. Structural indexing: Efficient 2d object recognition. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 12, 1198–1204. 33

Steinke, F., Schölkopf, B., & Blanz, V. 2005. Support vector machines for 3d shape processing. *Computer Graphics Forum 24*, 3, 285–294. 100

Sun, Y., & Abidi, M. 2001. Surface matching by 3d point's fingerprint. In *Proceedings of the Eighth IEEE International Conference on Computer Vision*, vol. 2, 263–269. 33

Sun, J., Yuan, L., Jia, J., & Shum, H.-Y. 2005. Image completion with structure propagation. *ACM Trans. Graph. 24*, 3, 861–868. 158

S.Yoshizawa, Belyaev, A. G., & Seidel, H. 2002. A simple approach to interactive free-form shape deformations. In *Pacific Graphics 2002 Proceedings*, 471–474. 136, 139

Tarel, J.-P., & Boujemaa, N. 1999. A coarse to fine 3d registration method based on robust fuzzy clustering. *Computer Vision and Image Understanding 73*, 1, 14–28. http://www-rocq.inria.fr/ tarel/cviu99.html. 33

Taubin, G. 1995. A signal processing approach to fair surface design. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM Press, 351–358. 101, 113

Turk, G. 2001. Texture synthesis on surfaces. In *SIGGRAPH 2001*, 347–354. 97

Verdera, J., Caselles, V., Bertalmio, M., & Sapiro, G. 2003. Inpainting surface holes. In *IEEE International Conference on Image Processing (ICIP 2003)*. 100

Wei, L.-Y., & Levoy, M. 2000. Fast texture synthesis using tree-structured vector quantization. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 479–488. 97, 108

Welch, W., & Witkin, A. 1992. Variational surface modeling. In *Computer Graphics (SIGGRAPH '92 Proceedings)*, E. E. Catmull, Ed., vol. 26, 157–166. 124, 127

Wood, D. N., Azuma, D. I., Aldinger, K., Curless, B., Duchamp, T., Salesin, D. H., & Stuetzle, W. 2000. Surface light fields for 3d photography. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 287–296. 56

Wyngaerd, J. V., & Gool, L. V. 2002. Automatic crude patch registration: toward automatic 3d model building. *Comput. Vis. Image Underst. 87*, 1-3, 8–26. 33

Yamany, S. M., & Farag, A. A. 2002. Surfacing signatures: An orientation independent free-form surface representation scheme for the purpose of objects registration and matching. *IEEE Trans. Pattern Anal. Mach. Intell. 24*, 8, 1105–1120. 33

Yu, Y., Zhou, K., Xu, D., Shi, X., Bao, H., Guo, B., & Shum, H.-Y. 2004. Mesh editing with poisson-based gradient field manipulation. *ACM Trans. Graph. 23*, 3, 644–651. 127

Zayer, R., Rössl, C., Karni, Z., & Seidel, H.-P. 2005. Harmonic guidance for surface deformation. In *Computer Graphics Forum, Proceedings of Eurographics 2005*, Blackwell, Dublin, Ireland, Eurographics. 127

Zeleznik, R. C., Herndon, K. P., & Hughes, J. F. 1996. Sketch: an interface for sketching 3d scenes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM Press, 163–170. 129

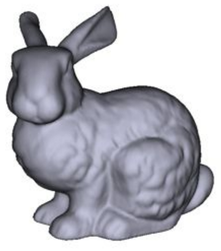
Zelinka, S., & Garland, M. 2004. Similarity-based surface modelling using geodesic fans. In *SGP'04: Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, Eurographics Association. 105

Zhu, S. C., Wu, Y., & Mumford, D. 1998. Filters, random fields and maximum entropy (frame): Towards a unified theory for texture modeling. *Int. J. Comput. Vision 27*, 2, 107–126. 97

Zorin, D., Schröder, P., & Sweldens, W. 1997. Interactive multiresolution mesh editing. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, T. Whitted, Ed., 259–268. 125

Zwicker, M., Pauly, M., Knoll, O., & Gross, M. 2002. Pointshop 3d: an interactive system for point-based surface editing. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, 322–329. 128

Data Sources



3D Scanning Repository, Stanford University, USA
http://graphics.stanford.edu/data/3Dscanrep/

3D Scanning Repository, Stanford University, USA
http://graphics.stanford.edu/data/3Dscanrep/

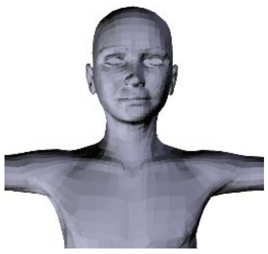
Digital Michelangelo Project, Stanford University, USA
http://graphics.stanford.edu/projects/mich/

Cyberware Inc., USA
www.cyberware.com

Cyberware Inc., USA
www.cyberware.com

Hugues Hoppe, Microsoft Research, USA
http://research.microsoft.com/~hoppe/

Polygon Technology GmbH, Germany
http://www.polygon-technology.com/



Poser 4 Model, e frontier Inc., USA
(The hand model depicted in chapter 8 is also part of this poser model (at different resolution))
http://www.e-frontier.com/

Roberto Scopigno, ISTI CNR Visual Computing Laboratory, Italy
http://vcg.isti.cnr.it/

MPI Saarbrücken, Germany
http://www.mpi-sb.mpg.de/departments/d4/

Yutaka Ohtake, MPI Saarbrücken, Germany
http://www.mpi-sb.mpg.de/~ohtake/mpu_implicits/

Curriculum Vitae

Publications

BENDELS, G. H., AND KLEIN, R. 2003. Mesh forging: editing of 3Dmeshes using implicitly defined occluders. In *Proceedings of the Eurographics/ ACM SIGGRAPH Symposium on Geometry processing*, Eurographics Association, 207-217.

BENDELS, G. H., KLEIN, R., AND SCHILLING, A. 2003. Image and 3Dobject editing with precisely specified editing regions. In *Vision, Modeling and Visualisation 2003*, Akademische Verlagsgesellschaft Aka GmbH, Berlin, T. Ertl, B. Girod, G. Greiner, H. Niemann, H.-P. Seidel, E. Steinbach, and R. Westermann, Eds., 451-460.

BENDELS, G. H., KAHLESZ, F., AND KLEIN, R. 2004. Towards the next generation of 3D content creation. In *Proceedings of the working conference on Advanced visual interfaces*, ACM Press, 283-289.

BENDELS, G. H., DEGENER, P., WAHL, R., KÖRTGEN, M., AND KLEIN, R. 2004. Image-based registration of 3D-range data using feature surface elements. In *5th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, Y. Chrysanthou, K. Cain, N. Silberman, and F. Niccolucci, Eds., Eurographics, 115-124.

BENDELS, G. H., KLEIN, R., SAMIMI, M., AND SCHMITZ, A. 2005. Statistical shape analysis for computer aided spine deformity detection. In *Journal of WSCG*, UNION Agency-Science Press, V. Skala, Ed., vol. 13, 57-64.

Bendels, G. H., Schnabel, R., and Klein, R., 2005. Fragment-based surface inpainting. *Poster proceedings of the Eurographics Symposium on Geometry Processing 2005*, July.

Rahimi, A., Keilig, L., Bendels, G., Klein, R., Buzug, T., Abdelgader, I., Abboud, M., and Bourauel, C. 2005. 3D reconstruction of dental specimens from 2D histological images and μCT-scans. *Computer Methods in Biomechanics and Biomedical Engineering 8, 3* (August), 167-176.

Bendels, G. H., Schnabel, R., and Klein, R. 2005. Detail-preserving surface inpainting. In *6th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, Eurographics, 41–48.

Müller, G., Bendels, G. H., and Klein, R. 2005. Rapid Synchronous Acquisition of Geometry and Appearance of Cultural Heritage Artefacts. In *6th International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)*, Eurographics, 13–20.

Bendels, G. H., Guthe, M., and Klein, R. 2006. Free-Form Modelling for Surface Inpainting. In *Proceedings of the 4th International Conference on Virtual Reality, Computer Graphics, Visualization and Interaction in Africa (Afrigraph 2006)*, ACM Press.

Bendels, G. H., Schnabel, R., and Klein, R. 2006. Detecting Holes in Point Set Surfaces. In *Journal of WSCG*, UNION Agency-Science Press, V. Skala, Ed., vol. 14, 89-96.